Resumen

Este Trabajo de Fin de Grado propone una metodología innovadora para optimizar calendarios de competición en equipos ciclistas profesionales mediante la combinación de redes de tipo Transformer y algoritmos evolutivos multiobjetivo (NSGA-II). El sistema busca maximizar la acumulación de puntos UCI y equilibrar la carga competitiva entre corredores a lo largo de la temporada. Los modelos predictivos, entrenados con datos históricos (2021-2025) de perfiles de etapas (GPX), atributos de ciclistas y resultados, alcanzan un 77.9% de precisión para predecir si un corredor finalizará en el *Top 50*. El algoritmo NSGA-II explora el espacio de soluciones generando un frente de Pareto. Las soluciones creadas incrementan la estimación de puntos UCI, así como la equidad respecto al calendario real del equipo entre enero y mayo de 2025. Aunque la validación empírica no confirma mejoras estadísticamente significativas frente a la planificación real, los resultados demuestran un alto potencial como herramienta de apoyo estratégico. El sistema identifica oportunidades en carreras subestimadas (p. ej. categorías 2.Pro/2.1) y prioriza corredores según su perfil. La solución, desarrollada con datos abiertos y técnicas reproducibles, ofrece una base escalable para la toma de decisiones basada en datos en contextos deportivos complejos.

Palabras clave: Optimización de calendarios, Algoritmos evolutivos, Ciclismo profesional, Aprendizaje automático.

Laburpena

Gradu Amaierako Lan honek txirrindularitza talde profesionalen lehiaketa egutegiak optimizatzeko metodologia berritzaile bat proposatzen du, Transformer sare neuronalak eta helburu anitzeko eboluzio-algoritmoak (NSGA-II) konbinatuz. Sistemak UCI puntu pilaketa maximizatzeko eta txirrindularien denboraldian zeharreko lehiaketa kargak orekatzeko helburuak ditu. Modeloen bidezko aurreikuspenak egiteko 2021 eta 2025 bitarteko lasterketa ibilbideen datu historikoak, txirrindularien ezaugarriak eta emaitza historikoak erabiltzen dira. Modeloek % 77,9-ko zehaztasuna lortu dute *Top 50a* aurreikusterako orduan. NSGA-II algoritmoak soluzioen espazioa ikertzen du, Pareto-ren fronte bat sortuz. Emaitzek txirrindularien lehiaketa kargen banaketa orekatzeaz gain, UCI puntuen aurreikuspena hobetu dute 2025eko urtarrila eta maiatza arteko egutegian. Balioztatze enpirikoak benetako plangintzaren aldean estatistikoki esanguratsuak diren hobekuntzak baieztatu ez dituen arren, emaitzek erakusten dute sortutako tresnak potentzial handia duela plangintza egiteko orduan laguntza estrategiko gisa. Sistemak egutegian sortu daitezkeen aukerak identifikatzen ditu gutxietsitako lasterketetan (adibidez, 2.Pro/2.1 kategorietan) eta haien profilaren arabera lehenesten ditu txirrindulariak. Proposatutako konponbidea datu irekiekin eta erreproduzitu ahal diren teknikekin garatu da, eta kirol testuinguru konplexuetan erabakiak hartzeko oinarri eskalagarria eskaintzen du.

Gako-hitzak: Egutegien optimizazioa, Eboluzio-algoritmoak, Txirrindularitza profesionala, Ikasketa automatikoa.

Abstract

This Final Degree Project proposes an innovative methodology to optimize competition calendars in professional cycling teams by combining Transformer neural networks and multi-objective evolutionary algorithms (NSGA-II). The system aims to maximize the accumulation of UCI points and to balance the competitive workload among riders throughout the season. The predictive models, trained with historical data (2021-2025) on stage profiles, rider attributes, and results, achieve 77.9 % Top 50 prediction accuracy. The NSGA-II algorithm explores the solution space by generating a Pareto front. The solutions improve the estimation of UCI points and enhance calendar equity compared to the actual calendar used between January and May 2025. Although empirical validation does not confirm statistically significant improvements over the team's actual calendar, the results

show high potential of the proposed solution as a strategic decision-support tool. The system identifies opportunities in underestimated races (e.g., 2.Pro/2.1 categories) and prioritizes riders based on their profiles. Developed with open data and reproducible techniques, this solution provides a scalable foundation for data-driven decision-making in complex sports contexts.

Keywords: Calendar optimization, Evolutionary algorithms, Professional Cycling, Machine Learning.

Índice de contenidos

Índice de figuras

Índice de tablas

Glosario y abreviaturas

Puntos UCI: Sistema de puntuación oficial utilizado por la Unión Ciclista Internacional (UCI) para clasificar a ciclistas, equipos y naciones en el ciclismo de carretera. Los puntos UCI se otorgan según los resultados obtenidos en competiciones reconocidas por este organismo, en función del tipo y categoría de la carrera, el puesto alcanzado y otros factores. Sirven para determinar rankings individuales y colectivos, y tienen implicaciones deportivas (clasificación mundial, acceso a carreras) y económicas (patrocinios, ascensos o descensos de división para los equipos).

Categorías de carreras: Las categorías de carreras en el ciclismo profesional siguen una nomenclatura particular. El primer número de la nomenclatura indica el tipo de carrera:

- 1.X → Carrera de un día
- 2.X → Vuelta por etapas

Por otro lado, el segundo número de la nomenclatura indica la categoría de la prueba, de mayor a menor en el siguiente orden:

- .UWT → .Pro → .1 → .2

Por lo tanto, una carrera 2.UWT es una carrera por etapas de máxima categoría, mientras que una carrera 1.2 es una carrera de un día de la mínima categoría.

API: Application Programming Interface

GA: Algoritmo genético (Genetic Algorithm)

ML: Aprendizaje Automático (Machine Learning)

MLP: Multilayer Perceptron. Durante el trabajo también se refiere como red neuronal densa.

PCS: Pro Cycling Stats

UCI: Union Cycliste Internationale

1. Introducción

La introducción por parte de la UCI de un sistema de clasificación con ascensos y descensos en el año 2023 ha intensificado la competencia y ha transformado la "lucha por los puntos" en un quebradero de cabeza para los directores deportivos. Los equipos ya no solo buscan victorias y exposición publicitaria, sino que están obligados a acumular puntos UCI para asegurar su permanencia en la división y obtener invitaciones a las competiciones más prestigiosas, incluidas las Grandes Vueltas.

Un calendario óptimo es fundamental para maximizar las prestaciones de los corredores a lo largo de la temporada. Tradicionalmente, la planificación del calendario ha sido una tarea artesanal, basada en la experiencia y la intuición de los directores deportivos.

No obstante, una buena planificación de calendario no solo implica distribuir las cargas de competición de forma equitativa, sino que también implica encontrar las alineaciones que permitan obtener la mayor cantidad de puntos. Para formar alineaciones óptimas es necesario considerar las características particulares de cada carrera, la cantidad de puntos a repartir y las cualidades de cada corredor en las carreras que mejor se le adaptan.

La complejidad de este nuevo contexto exige un enfoque estratégico y analítico en la planificación del calendario que pueda complementar la intuición y la experiencia de los directores deportivos. Para generar dicho enfoque analítico hacen falta herramientas que permitan analizar datos, predecir resultados y explorar sistemáticamente el amplio espacio de posibles combinaciones de carreras y corredores. Aquí es donde la gestión y el análisis de datos, combinados con técnicas de optimización avanzadas, pueden marcar una diferencia significativa.

Este Trabajo de Fin de Grado propone una solución innovadora a este problema: una red neuronal capaz de predecir resultados combinada con un sistema de optimización de calendario basado en algoritmos evolutivos. A diferencia del enfoque tradicional, este sistema permite una exploración sistemática y eficiente de posibles alineaciones y calendarios, buscando la combinación óptima que maximice los puntos UCI a la vez que distribuya equitativamente la carga de carreras de los corredores.

1.1 Estado del arte

Algoritmos de optimización en la planificación de competiciones deportivas

La planificación automatizada de calendarios deportivos es un problema complejo que ha sido ampliamente estudiado en las dos últimas décadas [1]. Los enfoques combinatorios incluyen programación entera mixta, programación por restricciones y diversas metaheurísticas a menudo híbridas para mejorar resultados [2]. Numerosas competiciones han sido programadas con dichas técnicas, desde ligas de fútbol hasta competiciones de hockey: por ejemplo, se han optimizado con éxito calendarios para la liga belga [3] y chilena de fútbol [4], la liga juvenil finlandesa de hockey sobre hielo [5] y eliminatorias sudamericanas de la FIFA [6], usando modelos matemáticos y heurísticos.

Otra línea de trabajo combina técnicas exactas y heurísticas. Por ejemplo, se presentó un enfoque pragmático para el *International Timetabling Competition 2021* [7], integrando movimientos heurísticos (intercambios de jornadas, rotaciones de emparejamientos, etc.) con modelos de programación por restricciones. Su metodología lograba horarios de alta calidad para torneos de “todos contra todos” a doble vuelta, incluso con instancias de gran tamaño, apoyado en un *solver CP-SAT*. En fechas más recientes, se propuso un esquema híbrido que combina una metaheurística evolutiva de biogeografía con un solucionador de programación entera. Aplicado al clásico *Travelling*

Tournament Problem, su método minimiza las distancias de viaje de los equipos en un calendario de liga, integrando iterativamente relajaciones parciales del modelo exacto integradas dentro del algoritmo metaheurístico [8]. Los resultados reportados mejoran sustancialmente soluciones previas, evidenciando cómo la "hibridación" de algoritmos evolutivos con técnicas exactas puede resolver problemas de calendario muy complejos de forma eficiente.

Cabe destacar que los algoritmos genéticos y evolutivos se han empleado también en la optimización de calendarios en distintas disciplinas [9], [10], [11]. Por ejemplo, se ha experimentado con un algoritmo genético para reorganizar el calendario de Fórmula 1, buscando reducir las distancias de viaje entre Grandes Premios. En un estudio de 2023, se comparó la solución obtenida mediante el "*add-on Solver*" de Excel contra un algoritmo genético implementado en MATLAB, demostrando la flexibilidad de los algoritmos genéticos para explorar secuencias de carreras alternativas [12]. En general, la literatura reciente muestra que los algoritmos de optimización permiten abordar la optimización multiobjetivo de calendarios con éxito. Estos avances sientan las bases metodológicas para problemas análogos en otros deportes de calendario complejo, como el ciclismo profesional.

Predicción del rendimiento deportivo con aprendizaje automático

Desde 2019, múltiples trabajos han aplicado aprendizaje automático (ML) para estimar tanto forma física como resultados, integrando grandes volúmenes de datos que incluyen estadísticas de competición, datos de sensores e historiales de rendimiento [13], [14] . Un ejemplo relevante es el trabajo de Sagi et al [15], donde se aborda por primera vez la asignación óptima de ciclistas a carreras mediante un enfoque basado en datos. En esta propuesta se entrenan clasificadores binarios, incluyendo métodos de boosting como CatBoost, que, dados los atributos de un ciclista y las características de una carrera, predicen de forma binaria la idoneidad de ese ciclista para participar en dicha prueba. Para ello se utilizan datos de entrenamiento recientes de plataformas como TrainingPeaks [16] y Strava [17] junto con propiedades de la carrera tales como distancia, desnivel y tipo de etapa. Este enfoque demuestra cómo el ML puede apoyar a los directores deportivos en la selección de corredores para cada evento. Las decisiones se basan en predicciones de rendimiento en lugar de la intuición o la popularidad histórica, alcanzando hasta un 80 % de precisión en la predicción de alineaciones [15].

Un estudio referente en el ámbito ciclista es el de Kholkine et al. [18], quienes desarrollaron un modelo de aprendizaje para ordenar (*learn to rank*) con el objetivo de predecir los diez primeros clasificados en 6 carreras de un día referentes en el calendario profesional. Utilizando datos históricos de ProCyclingStats (PCS) y características como la edad y el rendimiento previo de los ciclistas, entrenaron un modelo LambdaMART que superó en precisión a las predicciones realizadas por aficionados en plataformas especializadas.

En el ámbito del alto rendimiento olímpico, el uso de redes neuronales recurrentes ha mostrado beneficios tangibles. El Australian Institute of Sport (AIS) integró modelos de *Long Short-Term Memory* (LSTM) para pronosticar los *picos de forma* de sus atletas olímpicos, anticipando cuándo un deportista alcanzará su máximo rendimiento. Esta iniciativa logró aumentar en un 18% la tasa de medallas obtenidas en grandes competiciones, gracias a una mejor planificación del estado de forma con ayuda de las LSTM [19]. Este resultado ilustra cómo el uso de redes neuronales para la predicción del rendimiento futuro puede traducirse en ventajas competitivas concretas, permitiendo ajustar planes de entrenamiento y selección de eventos para optimizar el desempeño en el momento adecuado.

Uno de los pioneros en adoptar un enfoque data-driven en el ciclismo profesional fue el equipo NTT Pro Cycling (posteriormente conocido como Team Qhubeka). En 2019–2020, junto a su socio tecnológico NTT Ltd., desarrollaron una plataforma integral de datos para optimizar la planificación del calendario y la gestión de la plantilla. Esta herramienta ayudaba al equipo a *"seleccionar los*

corredores adecuados, preparar el calendario de carreras y asignar los corredores correctos a las pruebas" [20], con el objetivo de maximizar los puntos UCI y resultados obtenidos en la temporada. Según NTT, solo mediante la optimización algorítmica del calendario de competiciones se podría incrementar en un 20–30% la cantidad de puntos UCI obtenidos por el equipo. De hecho, los resultados iniciales fueron contundentes: a mediados de 2020, el equipo había logrado un 18% más de puntos que el año anterior, acompañado de un 200% más de victorias y 114% más de podios. Este caso real muestra cómo un sistema de apoyo a decisiones con algoritmos de optimización y predicción puede traducirse en ventajas competitivas en ciclismo, facilitando que directores deportivos diseñen calendarios óptimos.

Optimización de alineaciones en juegos tipo *fantasy*

Por último, un campo relacionado donde se requiere optimizar alineaciones y se aplican técnicas similares son los juegos tipo *fantasy* (ligas virtuales donde los aficionados eligen ciclistas reales para competir en base a puntos estadísticos). La optimización de alineaciones *fantasy* presenta un problema análogo al de seleccionar un equipo deportivo real, con restricciones de presupuesto, calendario, multiplicidad de jugadores y el objetivo de maximizar la puntuación esperada. En años recientes ha emergido literatura académica que aborda este problema con métodos de IA y optimización.

Mlcoch & Hubacek [21] revisaron la literatura existente hasta la fecha y propusieron un pipeline de simulación, que incluye varios componentes para capturar la incertidumbre y la correlación entre jugadores. Este pipeline tiene como objetivo generar muestras realistas del rendimiento conjunto de los jugadores con el fin de estimar la media de puntos esperados por jugador, la varianza individual y la covarianza entre jugadores. Después, se utiliza el *"Mixed Integer Quadratic Programming" (MIQP)* para seleccionar alineaciones con el objetivo de maximizar el puntaje esperado. Como resultado, aplicado a torneos semanales de la Premier League inglesa durante ocho semanas, el sistema generó un retorno de inversión superior al 34%.

2. Objetivos

En este apartado se definen el propósito principal del proyecto y los hitos intermedios que permiten alcanzarlo de forma ordenada y medible.

2.1 Objetivo general

Diseñar y validar una herramienta basada en algoritmos evolutivos y redes neuronales que, partiendo de datos históricos, caracterizaciones de ciclistas y recorridos, genere automáticamente calendarios de competición óptimos para un equipo ciclista profesional. Los calendarios propuestos han de ser capaces de proponer soluciones que maximicen la obtención de puntos UCI y garanticen una distribución equilibrada de la carga competitiva durante una temporada completa.

2.2 Objetivos específicos

- Recopilar los resultados oficiales de todas las pruebas masculinas de carretera disputadas entre 2021 y 2025, junto con los correspondientes ficheros *.GPX* y los atributos de los ciclistas integrantes del pelotón profesional.

- Preprocesar, limpiar y unificar las tres fuentes de datos, eliminando valores anómalos, homogeneizando unidades y versionando los conjuntos resultantes para asegurar la trazabilidad.

- Extraer y calcular variables geométricas y topográficas de los ficheros GPX y codificar otras variables categóricas relevantes.

- Entrenar dos redes neuronales profundas:

    - Una primera red entrenada específicamente para predecir el orden de llegada de corredores en carreras de un día y etapas, alcanzando al menos un *top-k accuracy* del 70% para el top 50.

    - Una segunda red capaz de predecir la clasificación final en vueltas por etapas integrando secuencialmente las características de las etapas que conforman la vuelta. Se pretende alcanzar al menos un *top-k accuracy* del 70% para el top 50.

- Implementar y calibrar un algoritmo genético (GA) que explore el espacio de calendarios posibles, maximizando la suma de puntos UCI y la equitativa distribución de los días de competición, tanto en el ámbito temporal como entre corredores.

- Evaluar retrospectivamente la herramienta en las temporadas 2024-2025, comparando los calendarios generados con el programa real del equipo y con calendarios aleatorios, así como analizar las fortalezas y limitaciones de la solución propuesta.

3. Metodología

Con el objetivo general de optimizar el calendario competitivo de un equipo profesional, maximizando tanto los puntos UCI obtenidos como la distribución en la carga de competición, la metodología se articula en cinco fases principales, recogidas en la Tabla 1.

FASES	OBJETIVOS ESPECÍFICOS	TAREAS	HERRAMIENTAS
Fase 1 Recopilación y auditoría de datos.	Compilar resultados históricos, ficheros GPX y atributos.	Recopilación de clasificaciones historicas	Python, requests, BeautifulSoup y pandas.
		Recopilación de ficheros GPX	
		Conversión *".cdb" → ".xml"*	
	Verificar consistencia y calidad.	Guardar un registro de calidad (data-log)	
Fase 2 Preprocesado.	Crear claves primarias a nivel ciclista y etapa para cruzar fuentes de datos.	Normalización de nombres.	Pandas, expresiones regulares y RapidFuzz.
		Fuzzy Matching.	
	Limpiar valores faltantes y atípicos	Imputación / filtrado de *outliers*	Pandas.
	Extraer características de perfil de etapa.	Generar variables a partir de los ficheros GPX.	Gpxpy.
Fase 3 Modelado predictivo.	Preparar datasets para entrenamiento.	Escalado y codificación de variables.	Scikit-learn y pytorch.
		Preparar tensores para carreras y GC.	Numpy y pytorch.
	Entrenamiento de las redes neuronales.	Entrenar diferentes arquitecturas de redes neuronales.	Pytorch, MLflow, optuna,XGBoost, GPU local y funciones personalizadas.
		Evaluar y comparar las arquitecturas.	
Fase 4 Optimización con GA (NSGA-II).	Diseñar el algoritmo de acuerdo con la problemática.	Establecer funciones objetivo y restricciones.	Pandas, numpy y scipy.
		Diseñar el cromosoma.	
	Diseñar operadores libres de conflictos.	Diseñar y probar operadores de cruce (crossover).	
		Diseñar y probar operadores de mutación.	
	Ejecutar NSGA-II y obtener frente de Pareto.	Crear el pipeline de ejecución del algoritmo.	
		Archivar el frente de Pareto y métricas clave para posterior visualización.	
Fase 5 Evaluación y visualización	Aterrizar la solución diseñada para la toma de decisiones empresariales.	Evaluar los calendarios optimizados respecto a los tradicionales	Numpy, matplotlib, seaborn.
		Crear figuras y dashboards que faciliten la interpretación de los resultados y la toma de decisiones	

Tabla 1. Definición metodológica del proyecto.

4. Desarrollo

Esta sección describe el proceso completo seguido para diseñar, entrenar y evaluar modelos capaces de predecir resultados a lo largo de todo el calendario, así como la posterior optimización mediante algoritmos evolutivos. Tal y como muestra la Ilustración 1 el proyecto se estructura en varios subapartados que cubren desde la recogida de datos hasta la obtención de los calendarios optimizados.

El proyecto se compone de dos grandes bloques funcionales:

i. **Bloque predictivo**: Modelos de aprendizaje automático que, a partir de variables de carrera, perfil y ciclista, estiman la posición esperada y, por extensión, los puntos UCI que cada corredor obtendría en una prueba concreta.

ii. **Bloque de optimización evolutiva**: Un algoritmo genético multiobjetivo (NSGA-II) que explora el espacio de calendarios posibles para maximizar (i) los puntos UCI agregados de la temporada y (ii) la equidad interna, respetando las restricciones establecidas.

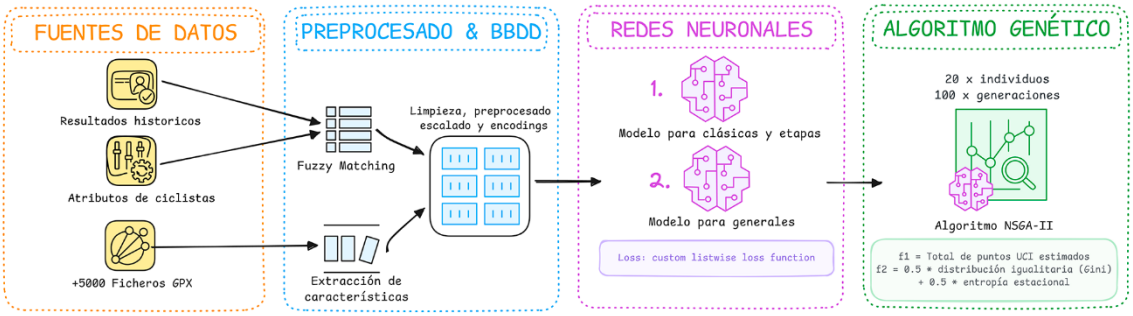

Ilustración 1. Flujo de trabajo de la herramienta creada. Elaboración propia.

4.1 Fuentes de datos

Tal y como se muestra en la Tabla 2 todos los datos utilizados para el entrenamiento de las redes neuronales son de origen público y libremente accesibles.

TIPO DE DATO	DESCRIPCIÓN	FORMATO
Resultados históricos	Clasificaciones de todas las pruebas de ciclismo masculino profesional (> 1.2/2.2). 2021-2025. También se han recopilado otros metadatos tanto de cada carrera (fecha y hora de la prueba, *startlist quality*, etc.) como de cada equipo (equipamiento utilizado cada temporada, categoría, etc.)	tabular
Atributos de ciclistas	Métricas básicas (edad, peso, altura) y 14 métricas de rendimiento categorizadas (Montaña, Sprint, resistencia, etc.). Las métricas son evaluadas según el criterio de un equipo de expertos que se dedica a ver todas las carreras del calendario. 2021-2025.	.cdb
Trazados de etapas	Ficheros del trazado de cada carrera de ciclismo masculino profesional (> 1.2/2.2) en crudo (latitud, longitud y elevación). 2021-2025.	.gpx

Tabla 2. Fuentes de datos.

Además de las fuentes principales, se ha calculado la latitud y la longitud medias de cada carrera, así como las coordenadas de salida y meta. Con estas variables y la fecha del evento se ha consultado la API de OpenWeatherMap [22]. La información obtenida ha permitido reconstruir las condiciones meteorológicas reales (temperatura, precipitación, viento, etc.). Conocer estos factores aumenta la

capacidad predictiva del modelo, puesto que los factores meteorológicos pueden realzar o penalizar el rendimiento de determinados ciclistas.

Sin embargo, dado que el objetivo es realizar predicciones con antelación de una temporada completa, se ha decidido descartar estas variables. Incluso empleando aproximaciones (por ejemplo, asignar el clima más habitual de la prueba) la incertidumbre añadida podría distorsionar en exceso los resultados.

Por otro lado, la disponibilidad de los perfiles detallados de recorrido, las catorce métricas que caracterizan las habilidades de cada ciclista y el historial íntegro de resultados permiten modelar prácticamente todas las variables relevantes conocidas antes del inicio de la competición. Aun así, el rendimiento de los modelos podría mejorar aún más incorporando datos de acceso más restringido tales como métricas de entrenamiento privadas o indicadores emocionales de estrés y motivación. No obstante, seguirían quedando fuera factores intrínsecamente impredecibles (averías mecánicas, pinchazos, caídas, enfermedades o decisiones tácticas inesperadas).

4.2 Preprocesado de los datos

4.2.1 Limpieza

El objetivo de la fase de limpieza ha sido garantizar que el conjunto de datos resulte consistente y completo, de modo que las redes neuronales puedan aprender con fiabilidad los patrones y característica propias de las clasificaciones de carreras ciclistas. Para ello se han llevado a cabo las siguientes tareas:

Preservación de variables temporales

a. La **hora de salida** se ha agrupado en tres categorías (mañana, tarde y noche) para mantener su relevancia como factor de rendimiento sin dispersión excesiva.
b. Puesto que la temporada ciclista comienza siempre en enero, la **fecha de la carrera** se ha convertido en un valor continuo que representa el *"día de la temporada"* (rango 1–365). De esta manera, el modelo capta posibles efectos estacionales o de progresión a lo largo de la temporada.

Tratamiento de valores ausentes o no clasificatorios.

Algunos ciclistas no obtienen una posición válida en la clasificación, lo que se refleja en resultados de clase *not a number* particulares de una carrera como *out of time limit (OTL)* o *did not finish (DNF)*. Dado que estas categorías no aportan información de orden que pueda aprenderse como variable objetivo, estos registros se han gestionado según los criterios especificados en la Tabla 3:

CASO	DESCRIPCIÓN	TRATAMIENTO
Outside time limit (OTL)	El ciclista cruza la línea de meta, pero con un retraso excesivo respecto al ganador, por lo que queda clasificado como fuera de control.	$max(posición) + 1$
Did not finish (DNF)	A diferencia del OTL, el ciclista no logra terminar la carrera. Esta variable presenta la desventaja de que no especifica en qué punto o bajo qué circunstancias se retira el ciclista. No es lo mismo abandonar al inicio que cerca del final, ni por fatiga que por una caída.	$max(posición) + n(OTL) + 1$
Did not start (DNS)	El ciclista no se presenta en la línea de salida. La opción más adecuada es tratarlo como si no hubiera participado en la carrera.	El ciclista se elimina del *dataframe* de la prueba.

Disqualified (DQS)	Ciclista descalificado por cometer algún tipo de infracción. Dado que esta circunstancia no nos permite conocer el desenpeño del ciclista estos casos han sido omitidos al igual que los *DNS*.	El ciclista se elimina del *dataframe* de la prueba.
No Result (NR)	Se asigna a todos los ciclistas de una carrera que ha sido suspendida. Como no se dispone de información sobre el desempeño de ninguno, se opta por eliminar completamente la prueba.	La prueba se elimina del *dataset.*

Tabla 3. Tipos de códigos "NAN" en una clasificación ciclista y su tratamiento.

En lo que respecta al resto de valores ausentes, el criterio para variables numéricas ha sido imputarlos utilizando la mediana de las carreras de la misma categoría del mismo país. Por otro lado, para variables categóricas desconocidas se ha creado una nueva categoría *"unknown"*.

Filtrado de carreras poco representativas

Para preservar la consistencia del modelo se han descartado las pruebas en las que finalizaron menos de 25 ciclistas, ya fuese por condiciones meteorológicas extremas o por tratarse de campeonatos nacionales con escasa participación.

4.2.2 Extracción de características GPX y *feature engineering*

Extracción de características GPX

En cada recorrido se ha automatizado el análisis en detalle de las características del trazado a partir de los ficheros GPX. Esto ha permitido enriquecer la base de carreras con métricas que reflejan la dureza y el perfil de cada etapa.

El primer paso de la extracción de características es calcular la distancia total en kilómetros y extraer estadísticas básicas de altitud (mínima, máxima, media y rango). En adición se calculan la latitud y longitud media, lo que permite situar geográficamente cada prueba. A continuación, se cuantifica la ganancia de elevación acumulada a lo largo de todo el perfil, lo que ofrece una medida global de exigencia.

Para capturar la dificultad de las subidas se ha diseñado un algoritmo que identifica los tramos continuos que superan un gradiente mínimo del 2%. Después se agrupan en puertos los tramos de subida detectados. Para cada puerto se calculan su longitud, desnivel y pendiente media.

Una vez se han calculado las características del puerto se genera un *"Score"* inspirado en el *Profile score de PCS* [23] para medir la dificultad. El Score de un puerto es

$$\text{Score}_{\text{puerto}} = \left(\frac{g}{2}\right)^2 \times L \times f(D),$$

donde g (%) es la pendiente media del tramo, L (km) la longitud del puerto y $f(D)$ un factor de atenuación según la distancia. Esta métrica pondera la longitud y la inclinación de cada subida, ajustado en función de la proximidad al final (más peso a las subidas decisivas). Finalmente, se suma el *"Score"* de todas las subidas para conocer el *Score* total de un segmento (p. ej. Últimos 25 km) o de la etapa completa.

Más allá de las subidas, también se obtienen métricas de segmento para los últimos 1, 5, 15 y 50 km: la ganancia total de elevación, la diferencia neta de altitud y la suma de los *"Score"* de las subidas contenidas en cada segmento. Estas variables ofrecen una visión detallada de cómo cambia la exigencia del recorrido según se acerca el final de la etapa. Por último, se calcula la proporción de distancia con pendiente elevada (por encima de umbrales como el 5 % u 8 %), lo que completa un conjunto de más de treinta atributos de trazado que destacan tanto la dureza global como los puntos clave de cada carrera (todas las variables están listadas en el anexo A).

Feature engineering

En paralelo, se han diseñado variables adicionales para capturar dinámicas de rendimiento histórico. Entre esas variables se encuentran las medias móviles de la posición de un ciclista en las últimas 3, 5 y 10 carreras o los días transcurridos desde la última competición.

No obstante, a pesar de que estas nuevas métricas ofrecen gran poder predictivo, no se han incluido en la versión final del modelo, pues en una simulación a varias carreras vista no se pueden conocer de antemano los resultados previos de cada corredor.

En cualquier caso, podría plantearse el uso de las predicciones de carreras anteriores como sustituto de los resultados reales. Sin embargo, esta estrategia introduciría un sesgo de dependencia en cadena: una predicción afectaría a la siguiente, y ambas influirían en las sucesivas. Este efecto acumulativo generaría una desviación progresiva a lo largo de la temporada, comprometiendo la validez y realismo de la simulación. La pérdida de eficacia a causa de eliminar estas variables se discute posteriormente en el apartado 5.1.

Por otro lado, se han generado dos nuevas variables objetivo que podrían resultar útiles como targets auxiliares o en futuras investigaciones, aunque no encajan directamente con la tarea de predecir los puntos UCI obtenidos (estos últimos dependen únicamente de la clasificación final). Por un lado, se ha calculado de manera continua el número de segundos de diferencia con respecto al ganador de cada etapa. Por otro, se ha categorizado la forma de victoria (p. ej. escapada en solitario, sprint reducido, contrarreloj) a partir de la descripción *"how won"* proporcionada por PCS.

4.2.3 Fuzzy Matching

El proceso de unificación de las distintas fuentes de datos ha implicado resolver inconsistencias en los nombres de ciclistas, equipos y carreras. Debido a que estos nombres pueden variar ligeramente según la fuente, ha sido necesario aplicar técnicas de coincidencia aproximada de cadenas de texto (*Fuzzy Matching*) para asegurar una correcta integración.

Normalización previa de nombres

Como paso previo al emparejamiento, todos los nombres han sido normalizados mediante una función específica que unifica patrones de escritura. Esta función se aplicó a nombres de corredores, equipos y carreras en todas las fuentes de datos relevantes, generando una versión "normalizada" de cada campo.

Asociación de ciclistas a su identificador único

Con el fin de asociar cada ciclista del conjunto a su correspondiente identificador único (*rider_id*), primero se han agrupado los corredores por equipo, y dentro de cada grupo se ha aplicado coincidencia aproximada entre los nombres normalizados. Se ha utilizado la métrica de similitud de la librería *RapidFuzz* [24], estableciendo un umbral de similitud del 80%. En caso de alcanzar dicho umbral, se ha asignado el *rider_id* correspondiente. Los casos sin coincidencias suficientemente confiables se han registrado en un archivo de salida para su posterior revisión manual.

Vinculación de carreras con trazados GPX

Una tarea especialmente crítica ha sido la asociación entre las carreras y los archivos de trazado GPX. Dado que ambos conjuntos usan nomenclaturas distintas para una misma carrera (p. ej. "Volta Ciclista a Catalunya" vs. "Volta a Catalunya"), se ha aplicado nuevamente una estrategia de coincidencia aproximada. En caso de no lograr una coincidencia exacta, se ha aceptado el emparejamiento con mayor puntuación siempre que esta haya superado el umbral de similitud.

Resolución jerárquica y validación

En ambos procesos de emparejamiento, se ha seguido una jerarquía de resolución que ha priorizado coincidencias exactas, luego coincidencias por normalización y finalmente *fuzzy matching*. Además, se han validado manualmente los nombres sin emparejamiento, asegurando un alto grado de precisión en la vinculación de entidades clave para el modelo.

El total de carreras para los que no se ha encontrado trazado GPX ha sido del 7.08%. No obstante, si se contabilizan carreras superiores a la categoría ".2" la cantidad desciende hasta el 1.17%.

4.3 Redes Neuronales

Los puntos UCI no se concentran únicamente en la meta de cada etapa: en las vueltas por etapas el grueso de la recompensa se adjudica según la clasificación general final. Dicha clasificación se calcula sumando los tiempos que cada ciclista emplea en cada una de las jornadas y, después, restando las bonificaciones y añadiendo las penalizaciones.

No obstante, predecir la clasificación general final plantea algunos retos adicionales respecto a predecir una clásica o una etapa. La principal diferencia es que, atendiendo a la forma de los datos de entrada, no basta con un tensor bidimensional $x \in \mathbb{R}^{N_{\text{riders}} \times N_{\text{features}}}$ donde cada fila describe al corredor en el contexto de esa etapa. Se necesita un tensor secuencial $x \in \mathbb{R}^{N_{\text{riders}} \times N_{\text{stages}} \times N_{\text{features}}}$ que preserve el orden cronológico y combine para cada ciclista sus atributos estáticos con las características específicas de cada etapa.

Escalado y codificación

Con el fin de acelerar el aprendizaje de las redes se ha aplicado un escalado estándar (media 0, desviación 1) a todas las características numéricas. Por otro lado, en lo que respecta a la codificación de variables categóricas (categoría de carrera, horario, etc.) se convierten en vectores *one-hot*. La lista completa de características utilizadas para el modelado se encuentra en el anexo A.

4.3.1 Justificación de las arquitecturas

Para decidir entre una red densa (MLP) y un modelo basado en Transformers debemos atender a ciertas diferencias listadas en la Tabla 4, además de dos características esenciales de las carreras por etapas:

1. **Interdependencia entre ciclistas**
    - MLP (Multilayer Perceptron): trata a cada corredor de forma aislada. Recibe su vector de atributos individuales y genera una predicción de posición sin "ver" directamente al resto del pelotón. [25]
    - Transformer: tal y como se ve en la Ilustración 2, gracias al mecanismo de *self-attention* [26], cada salida incorpora información de todos los corredores y del recorrido. Las múltiples cabezas de atención ponderan la relevancia de cada rival y del perfil de la etapa, posibilitando embeddings contextuales que reflejan fortalezas relativas.

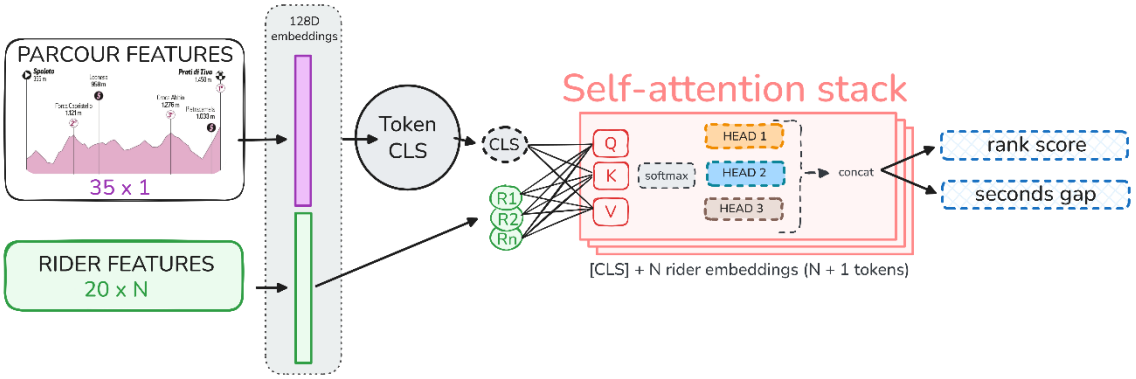

Ilustración 2. Diagrama de flujo para un Transformer multiobjetivo. Elaboración propia.

2. **Dinámica temporal de las vueltas por etapas**

   - Un MLP consume estadísticas agregadas (kilómetros totales, desnivel medio…) de todas las etapas sin explotar el orden cronológico de la prueba.
   - Un Transformer puede modelar explícitamente la secuencia de etapas (etapa *n* condiciona etapa *n+1*) si se le provee la serie temporal de características, aprovechando su estructura para capturar dependencias a lo largo del tiempo.

Característica	MLP (pointwise)	Transformer (listwise)
Visión del resto del pelotón	No - solo atributos propios	Sí - a través de *self-attention*
Orden cronológico	No - trabaja etapa a etapa de forma estática	Sí - incorpora dependencia secuencial entre etapas
Tipo de enfoque de ranking	Pointwise: estima posición de cada ciclista	Listwise: optimiza el orden global del pelotón
Flexibilidad en simulaciones	Alta — permite alterar inputs y reevaluar por corredor	Media - optimizada para el conjunto completo, menos adaptable individualmente

Tabla 4. Diferencias clave entre Transformers y MLP

4.3.2 Funciones de pérdida y métricas de evaluación utilizadas

Tradicionalmente, la mayor parte de la investigación sobre modelos de ranking se desarrolló en el ámbito de los motores de búsqueda, donde el objetivo es ordenar páginas web por relevancia [27]. Los mismos enfoques, tanto pointwise (MSE), pairwise (RankNet, LambdaRank) como listwise (ListNet, ListMLE), se han ido adaptando después a dominios tan diversos como la recomendación de productos [28], el filtrado de noticias [29] y, en este trabajo, la predicción de posiciones en ciclismo. Esta transferencia es posible porque todos comparten una estructura fundamental: dado un conjunto de ítems y un criterio de relevancia, aprender una función que los ordene de forma óptima según una métrica.

En este caso se han utilizado las métricas de la Tabla 5 como funciones de pérdida, mientras que las métricas de la Tabla 6 se han utilizado para evaluar la adecuación de la ordenación propuesta.

FINALIDAD	MÉTRICA Y DESCRIPCIÓN
Entrenamiento (Transformer)	**ListNet** Pérdida *listwise* que considera la lista completa de corredores como una distribución de probabilidad; entrena el modelo para que la distribución estimada reproduzca lo mejor posible el orden real. $$L = -\sum_{i=1}^{N} \frac{\exp(y_i / T)}{\sum_{j=1}^{N} \exp(y_j / T)} \ln\left(\frac{\exp(\hat{y}_i / T)}{\sum_{j=1}^{N} \exp(\hat{y}_j / T)}\right)$$ $T = 3$ *para no solo priorizar primeros puestos del ranking*
Entrenamiento (MLP)	**MSE (Error Cuadrático Medio)** Evalúa la diferencia promedio (al cuadrado) entre las posiciones reales y las posiciones predichas; cuanto menor es el valor, mejor se ajusta el modelo. $$MSE = \frac{1}{N}\sum_{i=1}^{N}(y - \hat{y})^2$$

Tabla 5. Funciones de pérdida utilizadas.

FINALIDAD	MÉTRICA Y DESCRIPCIÓN
Comparación de modelos	**Coeficiente de correlación de Spearman (ρ)** Mide la similitud entre el orden predicho y el orden real. Toma valor 1 cuando ambos rankings son idénticos, 0 cuando no existe relación y -1 cuando el orden está exactamente invertido. $$\rho = 1 - \frac{6\sum_{i=1}^{N} d_i^2}{N(N^2-1)}$$ $$d_i = rank\_pred_i - rank\_true_i$$
Comparación de modelos	**NDCG@K** Valora la calidad del ranking hasta la posición K, otorgando más importancia a los puestos altos y normalizando el resultado para que oscile entre 0 y 1 (donde 1 indica un ranking perfecto).
Comparación de modelos	**Top-K Accuracy** Porcentaje de corredores en los que las posiciones reales aparecen dentro de las primeras *K (1, 3, 10, 25, 50)* posiciones del ranking predicho. $$TopK = \frac{1}{N}\sum_{i=1}^{N} 1\{RankTrue_I \leq K\}$$

Tabla 6. Métricas de evaluación utilizadas.

4.3.3 Entrenamiento de los modelos

División temporal de los datos

Se ha aprovechado todo el horizonte temporal de los datos: 24 de enero de 2021 – 24 de mayo de 2025. Los conjuntos se han separado a través de una partición temporal a partir del 1 de junio de 2024.

- Entrenamiento: 200 327 registros correspondientes a 2 078 carreras (≈75% de los eventos).
- Prueba: 66 971 registros de 636 carreras posteriores en la línea temporal (≈25%).

Esta división temporal simula condiciones reales de predicción, donde los modelos deben generalizar a eventos futuros no vistos durante el entrenamiento.

Metodología de optimización de hiperparámetros

Para la búsqueda de hiperparámetros se ha aplicado optimización bayesiana mediante el *framework* Optuna [30], permitiendo una comparación justa entre las diferentes arquitecturas evaluadas. Esta metodología resulta más eficiente que la búsqueda por rejilla tradicional, especialmente para espacios de hiperparámetros de alta dimensionalidad. [31]

Los modelos se han evaluado exclusivamente en el conjunto de prueba para evitar fugas de información (*data leakage*) y obtener una medición realista de la capacidad de generalización. Para cada arquitectura se han optimizado los hiperparámetros específicos de su diseño, manteniendo consistencia en las métricas de evaluación.

Comparación de modelos

Las redes neuronales comparadas se describen en la Tabla 7. También se ha medido la eficacia de modelos de XGBoost y XGBoost optimizado para rankings. En adición se ha medido la eficacia de ordenar de forma aleatoria a los ciclistas como línea base con el fin de medir la mejora real de los modelos.

Comparar con un ranking aleatorio resulta útil, ya que cuando el *top-k* representa una gran parte del total de corredores a ordenar, es más fácil alcanzar un alto porcentaje de acierto incluso por azar. Por ejemplo, un 90 % de acierto en el *top-50* dentro de una *startlist* de 60 corredores puede parecer un rendimiento notable, sin embargo, al ordenar esos mismos corredores de forma completamente aleatoria se alcanzaría, en promedio, un acierto del 83 %.

MODELO	DESCRIPCIÓN
BaseTransformer	Transformer simple
MLP	Red neuronal densa sin mecanismos de atención. No ordena la lista de participantes, sino que predice el ranking esperado de cada ciclista.
MultiTaskingTransformer	Transformer entrenado para múltiples tareas: i. Predecir el ranking ii. Predecir a cuántos segundos del ganador llegará cada ciclista De esta forma se quiere comprobar si mediante la cuantificación del tiempo de retraso de cada ciclista se obtienen rankings más precisos.
Variantes "LessFeatures"	Estas variantes de los modelos, las cuales se han probado con el Transformer base y el MLP, no utilizan variables extrapoladas del histórico de resultados (p. ej. "posición media de las ultimas 3 carreras" o "días transcurridos desde el último día de competición"). De esta manera se sacrifica precisión a cambio de adaptar el modelo para que sea capaz de predecir a un año vista.

Tabla 7. Descripción de los modelos comparados.

4.4 Algoritmo Genético

El segundo bloque funcional del proyecto complementa los modelos predictivos mediante un algoritmo genético multiobjetivo NSGA-II [32]. Publicado por primera vez en 2002, este algoritmo optimiza simultáneamente varios criterios y obtiene un conjunto de soluciones óptimas: el frente de Pareto. Destaca por superar las carencias de las propuestas anteriores gracias, sobre todo, a su estrategia "elitista". En el presente trabajo, dicho algoritmo se ha utilizado para explorar el espacio de posibles calendarios y devolver al cuerpo técnico un frente de Pareto entre puntos UCI esperados y equidad de la carga competitiva.

4.4.1 Definición del problema

Los algoritmos genéticos (GA) son metaheurísticas inspiradas en la evolución natural darwiniana. Trabajan con una población de soluciones candidatas (cromosomas) que se recombinan (cruce) y mutan, guiadas por una función de aptitud (*fitness*). La selección de supervivientes favorece la supervivencia de los individuos más aptos, permitiendo escapar de óptimos locales y explorar regiones amplias del espacio de búsqueda. [11]

La búsqueda del calendario optimo se ha planteado como un problema de optimización multiobjetivo. Cuando el problema es multiobjetivo (maximizar los puntos UCI *y* mejorar la equidad de un calendario), no existe un único "mejor" valor de fitness: dos soluciones pueden ser mejores en distintos objetivos. Una posible alternativa es fusionar todos los objetivos en una única función agregada, por ejemplo $\mathcal{F} = w_1 \times \text{puntosUCI} + w_2 \times \text{equidad}$.

No obstante, esta aproximación presenta varios inconvenientes:

1. Ponderaciones a priori: Impone de entrada que un w_1 del énfasis esté en los puntos UCI y un w_2 en la equidad, aunque quizá existan soluciones interesantes en proporciones diferentes.

2. Pérdida de diversidad: Puede sesgar completamente la búsqueda hacia regiones del espacio que maximicen el valor agregado, dejando otras zonas inexploradas.

3. Decisión prematura: Obliga a definir unas prioridades fijas *antes* de conocer las soluciones factibles.

Para superar estas limitaciones, los algoritmos genéticos multiobjetivo (MOEA [33]) emplean el frente de Pareto, definido formalmente así:

> *Sea $\mathcal{P}$ un problema con $\mathcal{K}$ objetivos a optimizar. Una solución $\mathcal{S}_1$ domina a otra $\mathcal{S}_2$ si $\mathcal{S}_1$ es al menos tan buena en todos los objetivos y estrictamente mejor en al menos uno de ellos. El frente de Pareto es el conjunto de todas las soluciones no dominadas entre sí.*

Tal y como se puede ver en la Ilustración 3 cada punto en ese frente representa un compromiso factible entre los objetivos: no hay manera de mejorar simultáneamente en todos sin empeorar en alguno [34]. Por su parte, el conjunto de soluciones dominadas (representado con puntos azules) muestra aquellas soluciones que son mejoradas por otras en al menos uno de los dos objetivos.

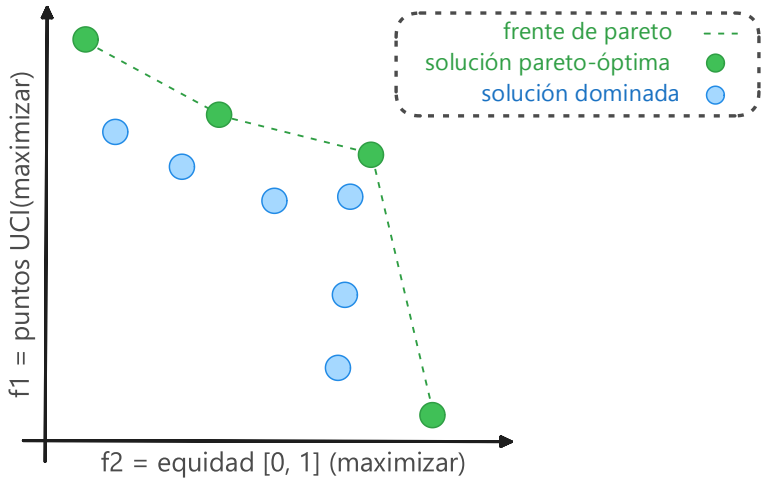

Ilustración 3. Frente de Pareto de una función con dos objetivos. Elaboración propia.

Este enfoque otorga varias ventajas:

- **Mayor diversidad de soluciones**
  En lugar de una única solución "óptima" según una función agregada, se obtiene un abanico de calendarios que recorren distintos balances entre puntos UCI y equidad.

- **Permite la toma de decisiones**
  El responsable directivo puede, tras la búsqueda, elegir el calendario que mejor se ajuste a sus necesidades reales (p. ej. priorizar equidad en si el contexto deportivo es favorable o puntos UCI si el ranking está muy reñido).

- **Evita la arbitrariedad de ponderaciones**
  No es necesario fijar de antemano grados de importancia, sino que las soluciones emergen de forma natural según la dinámica evolutiva.

- **Identificación de *knee points***
  En muchos casos, existen puntos del frente donde pequeños sacrificios en un objetivo suponen grandes ganancias en el otro. Localizar estos puntos ayuda a encontrar calendarios con una relación coste-beneficio especialmente atractiva para el responsable deportivo.

- **Facilidad de análisis y visualización**
  Al graficar los objetivos (ver Ilustración 3), el encargado de tomar las decisiones distingue de un vistazo el frente y las soluciones dominadas, apreciando huecos donde quizá convendría explorar más.

Funciones Objetivo

Sea $\mathcal{C} = \{c_1, \ldots, c_N\}$ el conjunto de carreras en los que participa un equipo y $\mathcal{R} = \{r_1, \ldots, r_M\}$ el conjunto de ciclistas que forman el equipo, un calendario se representa como un tensor binario:

$$\mathcal{S} \in \{0,1\}^{M\times N}, \mathcal{S}_{ij} = 1 \Rightarrow \textit{el ciclista } r_i \textit{ disputa la carrera } c_j$$

A partir de esta representación, se definen las dos funciones objetivo de elaboración propia que se detallan en la Tabla 8.

OBJETIVO	DESCRIPCIÓN	EXPRESIÓN
$\mathcal{F}_1 = \textbf{Puntos UCI}$	Suma de puntos esperados según las predicciones de las redes neuronales. Se agregan etapas y general para vueltas por etapas.	$\mathcal{F}_1(\mathcal{S}) = \sum_{i=1}^{M}\sum_{j=1}^{N} S_{ij}\hat{p}_{ij}$
$\mathcal{F}_2 = \textbf{Equidad}$	i. Igualdad global de días de competición D_i entre ciclistas, medida con el coeficiente de Gini G. ii. Distribución homogénea de esos días a lo largo de los K meses de la temporada, medida con la entropía normalizada H. Para cada ciclista $p_{i,m}$ es la fracción de sus días de competición que se disputa en el mes m.	$\mathcal{F}_2(\mathcal{S}) = \frac{(1-G(\mathcal{S}))+H(\mathcal{S})}{2}$, $G(\mathcal{S}) = \frac{\sum_{i=1}^{M}\sum_{k=1}^{M}\lvert D_i - D_k\rvert}{2M\sum_{i=1}^{M} D_i}$, $H(\mathcal{S}) = \frac{1}{M}\sum_{i=1}^{M}\frac{-\sum_{m=1}^{K} p_{i,m}\ln(p_{i,m})}{\ln K}$.

Tabla 8. Funciones objetivo del algoritmo genético multiobjetivo.

La función de equidad $\mathcal{F}_2(\mathcal{S})$ está diseñada específicamente para distribuir de forma justa los días de competición entre los ciclistas. Para ello combina dos métricas complementarias: el coeficiente de Gini $G(\mathcal{S})$, que mide la desigualdad en el número total de días de competición por corredor, y la

entropía normalizada $H(\mathcal{S})$, que evalúa cuán repartidos están esos días a lo largo de los meses de la temporada.

Minimizar $G(\mathcal{S})$ promueve que todos los ciclistas tengan un volumen de carga similar, evitando situaciones de sobreuso o subutilización. Maximizar $H(\mathcal{S})$, en cambio, favorece una distribución temporal equilibrada, lo que es relevante para mantener la forma física y prevenir acumulación de fatiga o largas inactividades.

Al combinar ambas métricas, se obtiene una medida robusta de equidad que no solo iguala las cargas totales, sino también su reparto temporal, algo esencial en deportes de resistencia como el ciclismo, donde la planificación progresiva y sostenida es clave. Por tanto, esta función resulta óptima para mejorar la asignación de días de competición desde una perspectiva tanto operativa (gestión del equipo) como fisiológica (rendimiento y recuperación).

Cabe recalcar que el diseño del proyecto es modular. Esto significa que el usuario puede modificar o añadir funciones objetivo. Por ejemplo, en un contexto en el que el patrocinador busque visibilidad, se puede añadir un tercer objetivo que busque maximizar el número de victorias o pódiums.

Restricciones impuestas

El tensor debe respetar ciertas restricciones, tanto reglamentarias como lógicas:

- Un máximo de 7 corredores alineados para cada carrera.
- Un mismo corredor no puede estar inscrito en más de una carrera para la misma fecha.

Existen dos formas de manejar estas limitaciones. Por un lado, pueden introducirse penalizaciones (o incluso una tercera función objetivo) que cuantifiquen las violaciones. Por otro, pueden generarse calendarios iniciales factibles y utilizar operadores de cruce y mutación que conserven dichas restricciones. Finalmente, tras probar el enfoque con penalizaciones, no se ha conseguido reducir los solapamientos por debajo de 50 tras 100 generaciones. Por lo tanto, se ha adoptado la segunda alternativa, la cual garantiza la factibilidad desde el primer instante.

Además, este enfoque reduce el coste computacional respecto al método basado en penalizaciones. Esto se debe a que, en el enfoque penalizado, el cálculo de la función objetivo implica revisar el calendario completo de cada individuo para detectar y cuantificar los solapamientos. Además, la convergencia es más lenta, ya que el algoritmo debe simultáneamente mejorar los dos objetivos originales (puntos UCI y distribución de carga) y, de forma indirecta, aprender a minimizar los solapamientos para reducir la penalización. Esta dificultad añadida puede requerir más generaciones para alcanzar soluciones viables, elevando así el coste computacional total.

En cambio, los operadores propuestos en el apartado 4.4.2 sólo verifican la disponibilidad de los ciclistas al crear descendientes, sin necesidad de cálculos adicionales durante la evaluación.

Finalmente, cabe destacar que en un futuro existe la posibilidad de agregar restricciones internas que pueda establecer el equipo. Algunos posibles ejemplos son:

- Objetivos estratégicos específicos
- Períodos mínimos de descanso entre pruebas.
- Participación de ciertos corredores en carreras de su región.
- Minimización de costes logísticos.

4.4.2 Operadores

En cada generación t, a partir de la población actual P_t (padres), se aplica selección, cruce y mutación para generar una población de descendientes Q_t del mismo tamaño. No obstante, en lugar de descartar inmediatamente a los padres, se forma una población intermedia $R_t = P_t \cup Q_t$ la cual tiene el doble de individuos ($2 \times N$).

Una vez evaluadas todas las soluciones de la nueva población, los individuos se clasifican por frentes de Pareto $F_1, F_2, \ldots$ mediante *non-dominated sorting* (ver Ilustración 4).

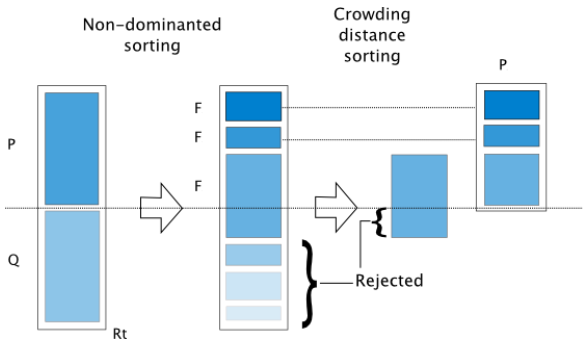

Ilustración 4. Proceso de selección por frentes. Fuente: *pymoo*.

Al construir la siguiente población P_{t+1} de tamaño N, se añaden primero todos los miembros de F_1. Si la suma de los frentes completos hasta F_{k-1} es menor que N pero F_k los supera se toman solo los mejores individuos de F_k ordenados por *crowding distance.*

Este criterio garantiza la diversidad dentro del frente: a los extremos (por ejemplo, S_1 y S_4) se les asigna distancia infinita para preservarlos, y entre S_2 y S_3 sobrevive el que crea un "espacio" mayor con sus vecinos (ver Ilustración 5). De este modo, las mejores soluciones siempre se conservan íntegramente (enfoque elitista) y, al mismo tiempo, se exploran con amplitud las posibles soluciones.

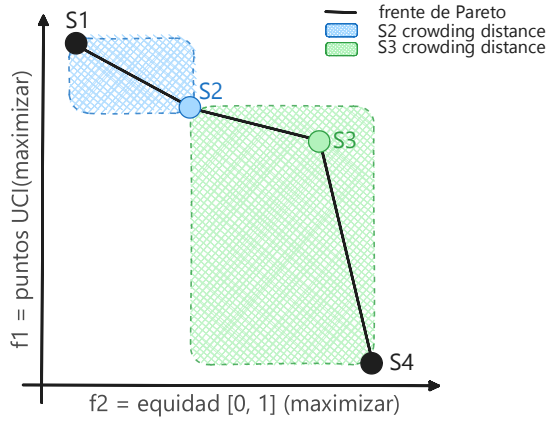

Ilustración 5. Selección por *crowding distance*. Elaboración propia.

Operador de cruce (*Crossover*)

Cada progenitor se elige mediante un torneo binario sobre la población actual:

1. Se extraen dos índices al azar.
2. Gana el que esté en un frente de Pareto mejor.

3. Si empatan en el mismo frente, decide la distancia de *crowding*. Sobrevive el individuo más "aislado" y, por tanto, más valioso para la diversidad.

La probabilidad de ejecutar el operador de cruce es del 90%, por lo contrario, se copia íntegramente uno de los padres al azar (A o B). Si se está dentro del 90 % anterior, para cada carrera se decide con 50% si se hereda la máscara de corredores del padre A o la del padre B.

En el individuo hijo solo se copian aquellos corredores que están libres de conflictos de fechas, es decir, los que ocasionarían solapamientos se descartan. Para ello, antes de transferir cada bloque de siete corredores se comprueba, uno a uno, si el ciclista está disponible para las fechas de la carrera destino. Finalmente, sólo los que pasan el filtro se copian.

Si tras el filtro la carrera queda con menos de siete corredores, se rellenan los huecos escogiendo al azar compañeros que sigan libres. De esta forma, al finalizar, todas las carreras conservan exactamente siete corredores y ningún ciclista aparece en dos pruebas solapadas, cumpliendo las restricciones sin penalizaciones adicionales. La Ilustración 6 ejemplifica la problemática en un escenario de 3 corredores por carrera.

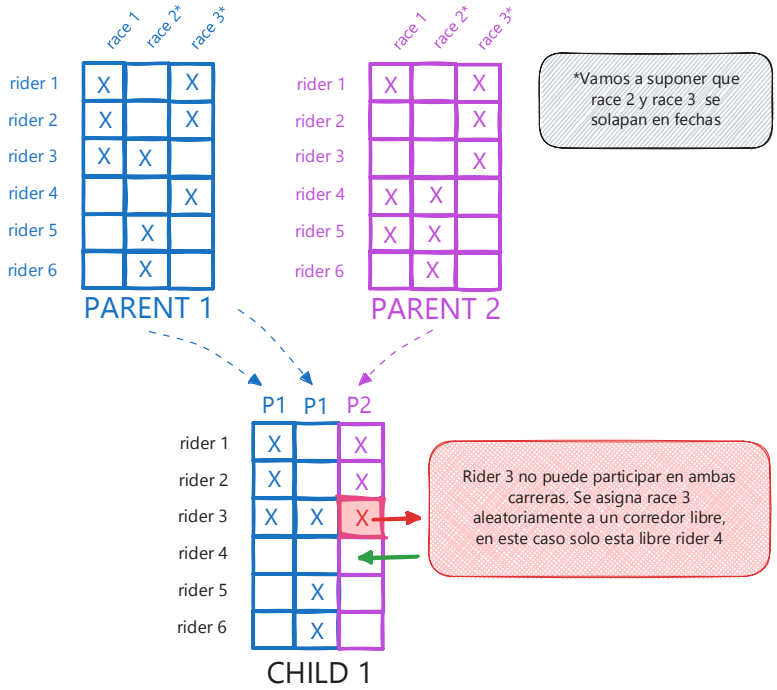

Ilustración 6. Ejemplo de cruce simplificado. Elaboración propia.

Operador de mutación

Después de crear el descendiente mediante el cruce, se sortea por separado si se aplica alguno de los dos operadores de mutación que introducen variación sin violar las restricciones de factibilidad.

a. Intercambio dentro de la propia carrera ($p_{within} = 0.2$):

- Se escoge al azar uno de los siete corredores alineados y se intenta sustituirlo por un compañero que no tenga solapamientos con esa prueba. Se verifica la disponibilidad fecha a fecha. El cambio sólo se confirma si el nuevo corredor supera esta comprobación. De esta manera se generan ajustes tácticos finos manteniendo intacto el cupo de siete ciclistas. La Ilustración 7 ejemplifica la problemática en un escenario de 3 corredores por carrera.

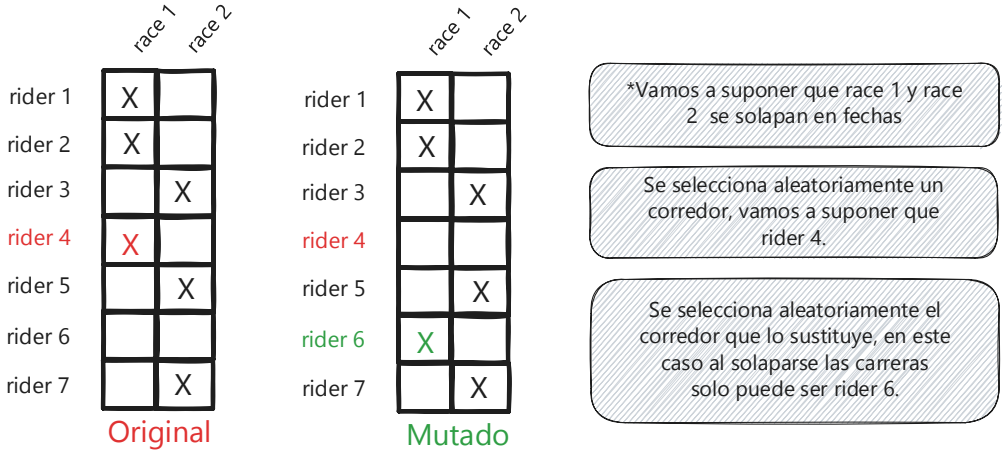

Ilustración 7. Ejemplo de mutación en carrera. Elaboración propia.

b. Intercambio de alineaciones entre dos carreras distintas ($p_{between} = 0.1$):

- Se escogen dos carreras al azar. Se barajan las alineaciones de ambas y se exploran pares de corredores (uno por carrera) hasta encontrar la primera combinación tal que, tras cruzarlos, ambos sigan libres de solapamientos. Si existe ese par, se realiza el trueque y la función termina. Si no, el calendario permanece tal cual. La Ilustración 8 ejemplifica la problemática en un escenario de 3 corredores por carrera.

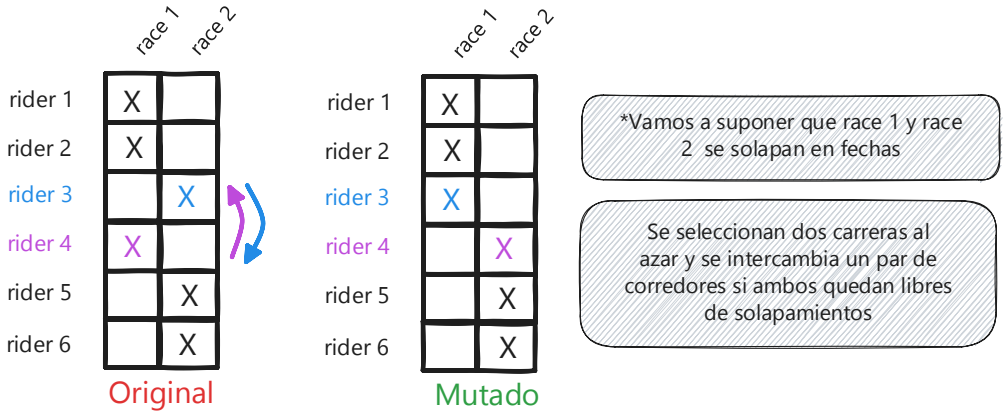

Ilustración 8. Ejemplo de mutación entre dos carreras. Elaboración propia.

- Este movimiento produce saltos más amplios en el espacio de soluciones, redistribuyendo fuerzas sin quebrar la coherencia global.
- La búsqueda local cuesta ≤ 49 intentos (7 × 7) y solo ocurre en el 10 % de los casos, por lo que el impacto sobre la complejidad total es O(R) (recorrer el calendario).

A continuación, se muestra en la Ilustración 9 el diagrama de flujo completo de la implementación de NSGA-II que se ha descrito:

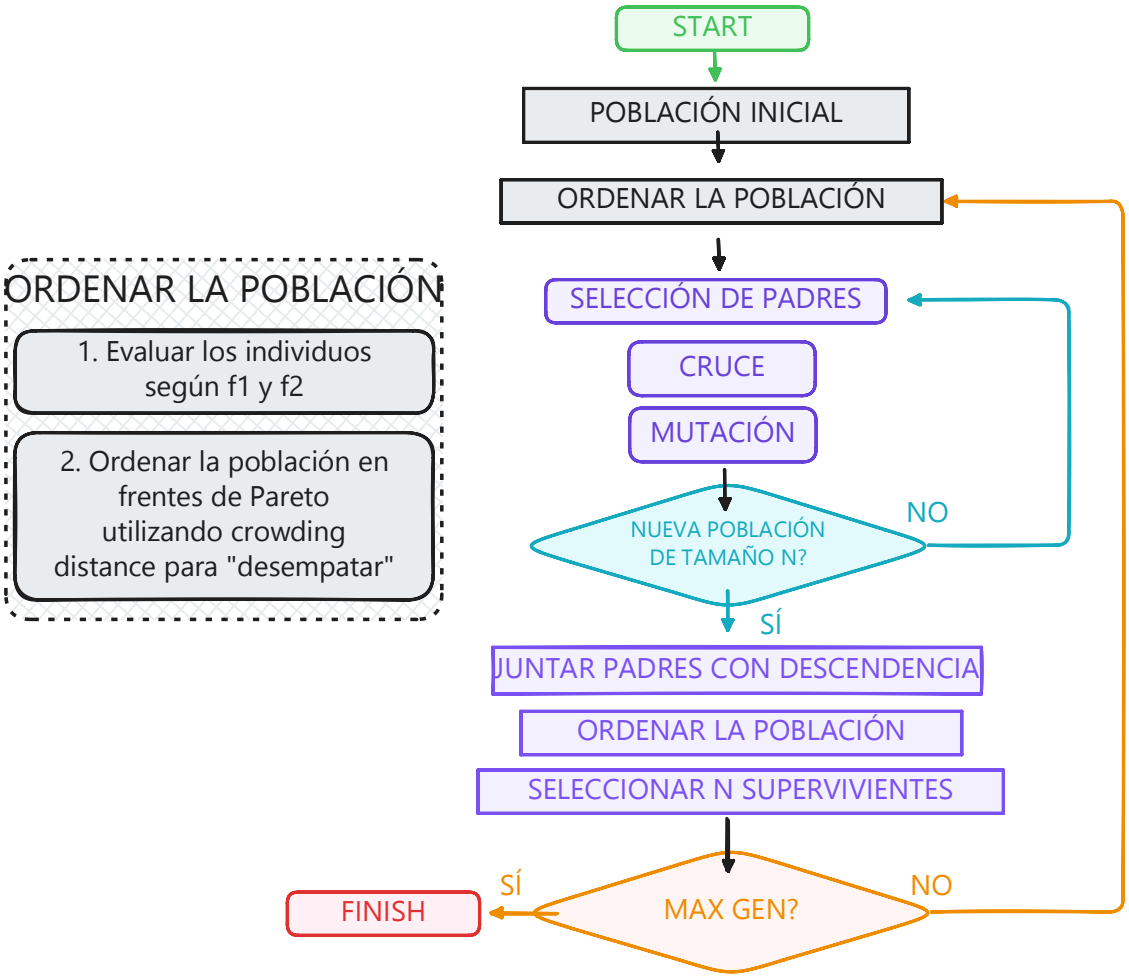

Ilustración 9. Arquitectura y flujo del (Algoritmo genético) GA utilizado. Elaboración propia.

5. Resultados

En este apartado se presentan los hallazgos empíricos más relevantes derivados, por un lado, del proceso de entrenamiento y ajuste de los modelos de aprendizaje automático y, por otro, del proceso de optimización mediante NSGA-II.

5.1 Resultados del entrenamiento de modelos

Para cada modelo candidato, en primer lugar, se ha llevado a cabo una búsqueda exhaustiva de hiperparámetros con el objetivo de disponer de la versión óptima de cada uno. En el caso de la red neuronal Transformer en su arquitectura base, la Ilustración 10 presenta de forma detallada cómo varía la métrica NDCG@25 al ajustar dimensiones del modelo.

La Ilustración 10 demuestra cómo Optuna concentra los experimentos en las regiones más prometedoras. Por ejemplo, al comprobar que valores bajos de *dropout* mejoran el rendimiento, enfoca los siguientes intentos en ese rango. Este proceso ha combinado aislar la combinación de hiperparámetros que maximiza la capacidad predictiva: dimensiones de *embedding* amplias, *dropout* moderado y una temperatura elevada de ListNet la cual enfoca el descenso de gradiente más allá de las primeras posiciones.

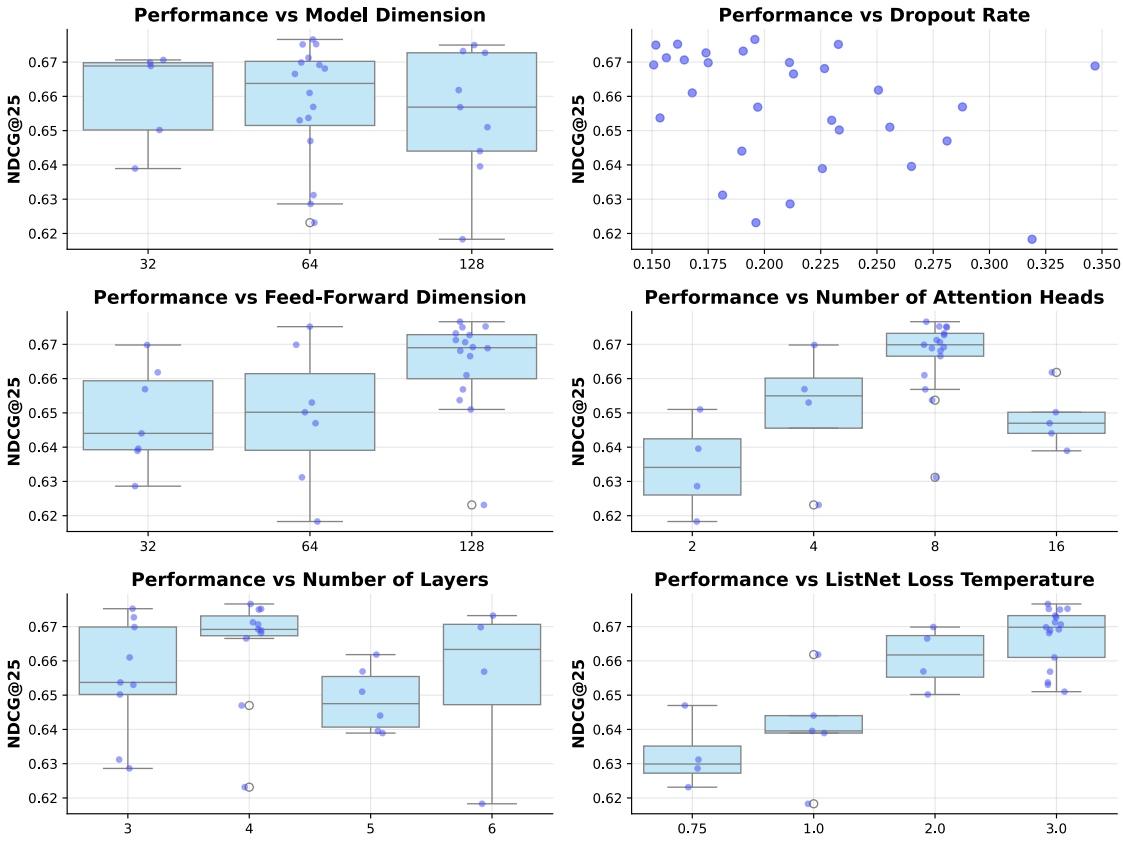

Ilustración 10. Visualización de la búsqueda de hiperparámetros óptimos. Elaboración propia.

Tras fijar los hiperparámetros óptimos, se ha comparado el rendimiento de cada variante en diversas métricas de rendimiento. La Tabla 9 muestra que la arquitectura Transformer en su configuración base alcanza los valores más altos en 6 de los 9 indicadores, superando tanto a los métodos de *boosting* como a las redes MLP. Además, aunque el Multi-Tasking Transformer introduce la variable auxiliar de diferencia temporal y mejora ligeramente algunas métricas (p. ej. NDCG@25 = 0.642

frente a 0.639 del base), no ofrece una ventaja global sostenida sobre la arquitectura base. Con ello se confirma que los mecanismos de atención constituyen la estrategia más eficaz para este problema.

	Random	XGBoost	Ranking XGBoost	Transformer Base	Transformer Base*	MLP	MLP*	Multi-Tasking Transformer
ndcg@5	0.053	0.191	0.197	0.221	0.179	0.199	0.165	**0.222**
ndcg@10	0.107	0.315	0.339	**0.379**	0.299	0.334	0.291	0.376
ndcg@25	0.283	0.552	0.573	0.639	0.527	0.569	0.522	**0.642**
Acc top 1	0.007	0.119	0.088	0.168	0.195	**0.197**	0.182	0.138
Acc top 3	0.030	0.234	0.192	**0.300**	0.291	0.286	0.278	0.248
Acc top 5	0.056	0.303	0.259	**0.388**	0.337	0.345	0.335	0.323
Acc top 10	0.109	0.402	0.369	**0.514**	0.419	0.442	0.419	0.451
Acc top 25	0.286	0.576	0.569	**0.677**	0.559	0.597	0.558	0.642
Acc top 50	0.486	0.698	0.706	**0.779**	0.678	0.711	0.679	0.766

*Variante sin *features* de resultados recientes

Tabla 9. Comparación de resultados de los modelos óptimos.

Para mostrar de forma más intuitiva estas diferencias, la Ilustración 11 muestra el *accuracy* en función de k (1, 3, 5, 10, 25 y 50) y el promedio total, reflejando el salto de rendimiento que ofrecen los modelos basados en Transformers.

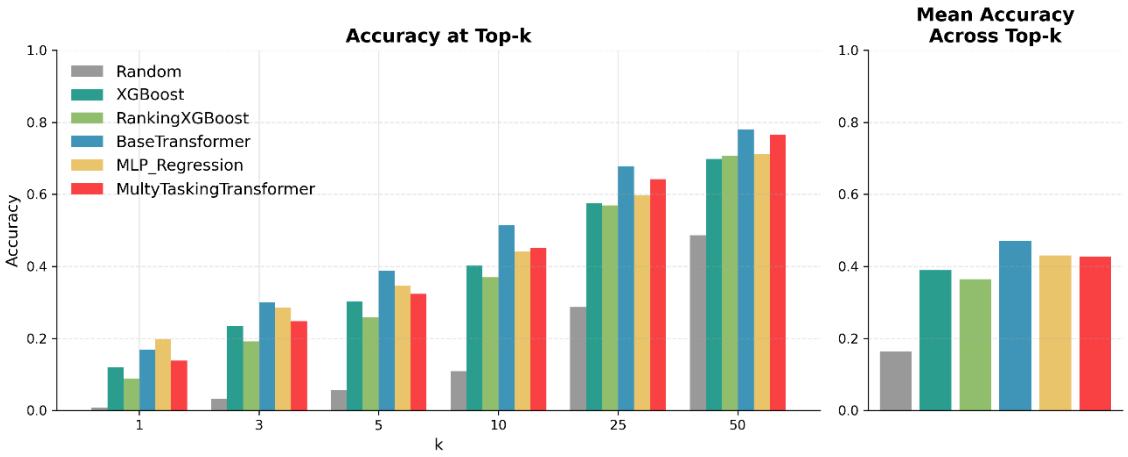

Ilustración 11. Comparativa de rendimiento para los modelos. Elaboración propia.

Para cuantificar el impacto de reducir las variables, en la Ilustración 12 se representa la ganancia de *accuracy* sobre el método aleatorio (eje y) cuando se pasa de la versión completa a la reducida (*less features*). Tanto el Transformer como la MLP experimentan descensos graduales. De esta forma se evidencia la importante capacidad predictiva de las variables de componente temporal, las cuales tienen la capacidad de captar las tendencias temporales en el estado de forma de los ciclistas. No obstante, las versiones reducidas mantienen un poder predictivo considerable respecto a la línea base establecida.

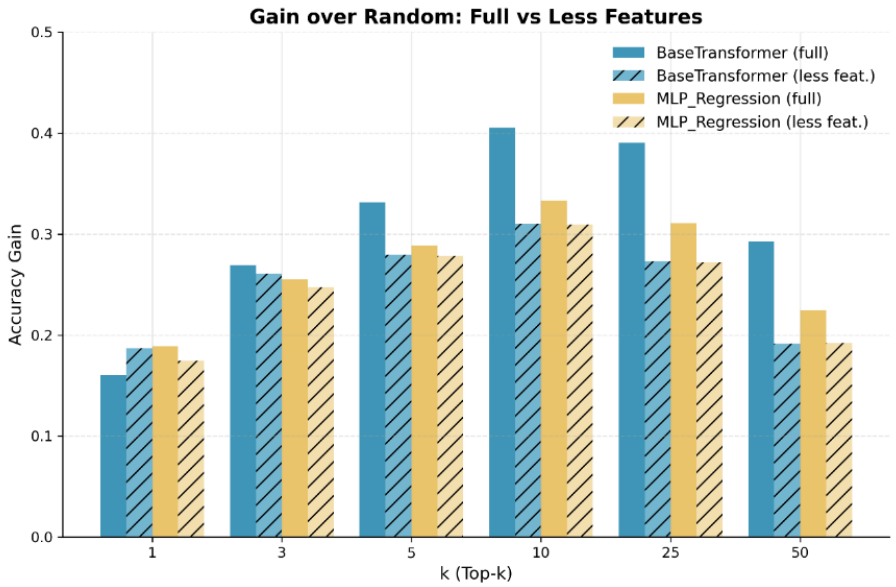

Ilustración 12. Pérdida de rendimiento con menos características. Elaboración propia.

Finalmente, la Ilustración 13 ofrece el histórico de la función de pérdida durante el proceso de búsqueda de hiperparámetros para la arquitectura Transformer, evidenciando la convergencia estable del modelo óptimo tanto en entrenamiento como en validación a lo largo de las épocas. Con esta metodología, se ha asegurado no sólo la obtención de las configuraciones más efectivas, sino también la fiabilidad y reproducibilidad de los resultados presentados.

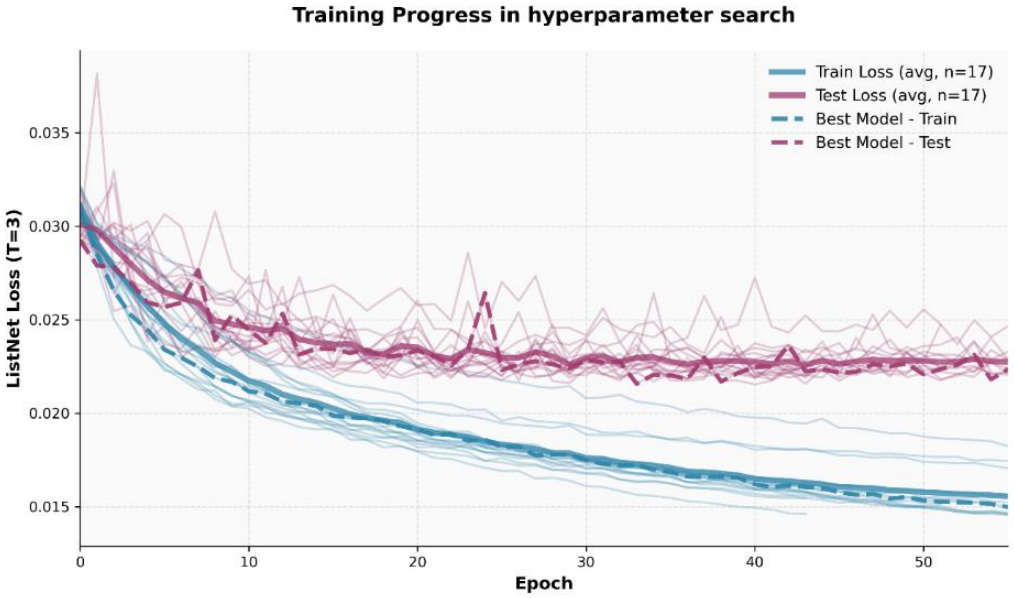

Ilustración 13. Histórico de la función de pérdida. Elaboración propia.

5.2 Resultados de los algoritmos evolutivos

Con el fin de medir la efectividad de la optimización realizada por el algoritmo NSGA-II se ha comparado su desempeño con un algoritmo evolutivo más simple. Por otro lado, se ha comparado el calendario optimizado con el calendario real del equipo y un calendario aleatorio sin optimizar.

5.2.1 Efectividad de NSGA-II vs GA simple

El GA base sigue el flujo clásico "evaluar → seleccionar → variar → reemplazar". No guarda memoria entre generaciones: la población hija sustituye por completo a la anterior. Esto implica que si un buen calendario no es elegido padre puede perderse definitivamente.

El NSGA-II introduce dos engranajes extra: (1) clasificación rápida en frentes no dominados para determinar ganadores en selección de padres mediante torneo y (2) elitismo ambiental que combina padres + hijos antes de filtrar. Así, cada generación mantiene una jerarquía de niveles y selecciona primero por nivel de frente de Pareto y luego por *crowding*. De esta forma se conservan siempre los calendarios más prometedores y se distribuye la población a lo largo de todo el frente de Pareto.

Ambos algoritmos se han probado con una población de 20 individuos durante 100 generaciones.

Convergencia de los objetivos

La Ilustración 14 muestra en una ejecución para un equipo UCI ProTeam la evolución simultánea de los puntos UCI (línea sólida) y su media poblacional (línea discontinua). NSGA-II supera al GA antes de la generación 15 y finaliza con 2591 pts frente a 2315 pts (+11.9 %).

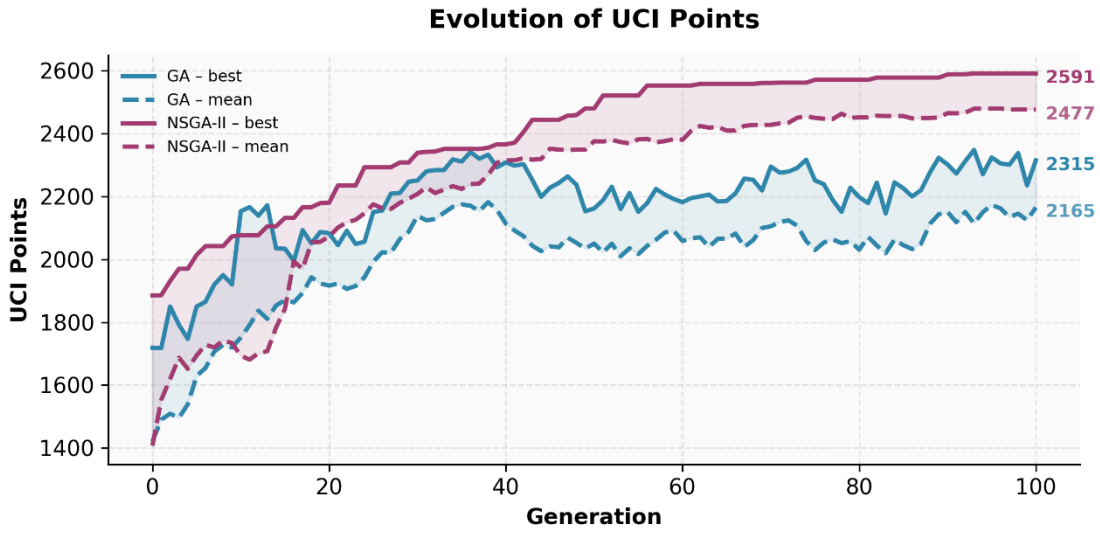

Ilustración 14. Convergencia en optimización de puntos UCI. Elaboración propia.

Asimismo, la Ilustración 15 refleja un patrón análogo para la equidad: 0,9115 vs 0,8824 (+3.3 %). Por otro lado, la efectividad de la selección elitista queda de manifiesto, puesto que, aunque en etapas especificas ambos algoritmos tienen rendimiento similar, la preservación de los mejores individuos permite una búsqueda de óptimos más precisa que otorga mejores resultados a NSGA-II.

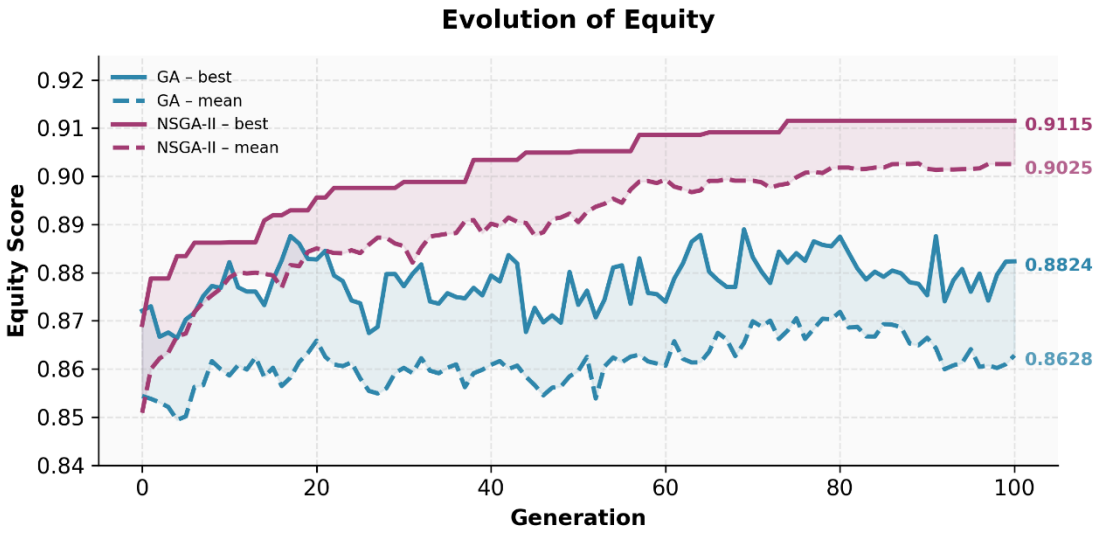

Ilustración 15. Convergencia en optimización de equidad. Elaboración propia.

Riqueza del frente de Pareto

En la Ilustración 16 NSGA-II sostiene mayor cantidad de calendarios no dominados por generación y termina con 11, frente a los 3 del GA. Además, alrededor de la generación 40, cuando el número de soluciones cae NSGA-II vuelve a generar alternativas no dominadas, recuperando la diversidad. En términos prácticos este comportamiento se alinea con la estrategia de elitismo por frentes y garantiza más alternativas de calendarios al cuerpo técnico.

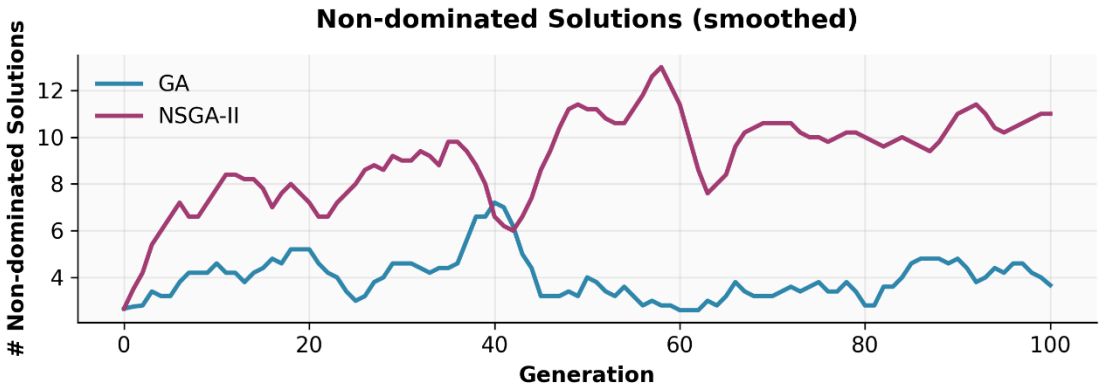

Ilustración 16. Cantidad de soluciones no dominadas por generación. Elaboración propia.

6. Conclusiones

Este Trabajo de Fin de Grado ha demostrado que la combinación de técnicas avanzadas de *machine learning* (redes Transformer) y optimización multiobjetivo (NSGA-II) permite diseñar calendarios de competición que maximizan la obtención de puntos UCI y equilibran adecuadamente la carga competitiva en equipos de ciclismo profesional. Este enfoque complementa un proceso tradicionalmente intuitivo con un método reproducible y basado en datos objetivos.

6.1 Aportaciones principales

- **Modelo predictivo puntero**: El Transformer básico implementado ha alcanzado un rendimiento destacable, logrando un NDCG@25 de 0.639 y una precisión en el top-50 del 77.9 %. Si bien la predicción no ha sido el único foco de este trabajo, el modelo demuestra un alto nivel de precisión que lo hace potencialmente extrapolable a otros contextos. Potenciales usos pueden ser la predicción de resultados, la identificación de rivales relevantes antes de una carrera o el análisis de sensibilidad ante cambios en el recorrido o las condiciones climáticas.

- **Estrategia evolutiva eficaz**: La implementación del algoritmo NSGA-II ha permitido generar un frente de Pareto compuesto por hasta once calendarios no dominados. De esta manera se ha mejorado significativamente (+11.9%) la cantidad total de puntos UCI obtenidos en comparación con GA básicos que no implementan una estrategia elitista. Al mismo tiempo, se ha encontrado un equilibrio más eficiente en la distribución de la carga competitiva (+3.3%).

- **Validación práctica**: Se ha identificado que el modelo reserva los corredores más fuertes para aquellas competiciones donde el coste por punto es menor, sugiriendo un potencial incremento en la eficiencia del calendario al centrarse en pruebas 2.Pro y 2.1 estratégicamente seleccionadas.

- **Pipeline reproducible**: El desarrollo del proyecto se ha basado íntegramente en datos abiertos, facilitando su aplicación potencial en otros equipos ciclistas profesionales o en contextos deportivos diferentes.

Estas contribuciones proporcionan un valor significativo tanto en el ámbito académico como en el profesional, ofreciendo nuevas herramientas que permiten optimizar decisiones deportivas estratégicas.

En este trabajo se busca subrayar cómo la inteligencia artificial puede complementar eficazmente la experiencia humana, ofreciendo herramientas adicionales que enriquecen la toma de decisiones estratégicas sin sustituir el criterio experimentado que solo los profesionales deportivos pueden aportar.

6.2 Limitaciones

Sin embargo, también es importante reconocer algunas limitaciones identificadas. Al planificar un calendario a un año vista, se desconocen los perfiles y las *startlist* de la mayoría de las carreras. Por lo tanto, es obligatorio asumir que estos serán parecidos a los de los años anteriores.

Además, las predicciones realizadas por la red neuronal pueden resultar excesivamente optimistas, como se ha visto en el caso del Tour de Hainan. Esta casuística se da particularmente en carreras de menor nivel, afectando la precisión final de los calendarios generados.

Por otro lado, este tipo de metodologías predictivas dejan fuera factores externos como lesiones, condiciones meteorológicas o aspectos logísticos que pueden alterar considerablemente los resultados previstos.

6.3 Líneas futuras

Como posibles líneas futuras de investigación, se propone la automatización avanzada en la extracción de características GPX mediante técnicas no supervisadas como *autoencoders* [35], [36], lo que permitiría caracterizar los recorridos de forma más rica y flexible. Además, se plantea enriquecer el modelo predictivo incorporando datos adicionales procedentes de plataformas como Strava [17], lo cual podría abrir la puerta a modelar información relevante sobre la carga de entrenamiento y el estado de forma de los corredores. Por otro lado, convendría refinar el optimizador para integrar variables como los picos de forma planificados de antemano, los objetivos personales de cada ciclista o la necesidad de acumular ritmo competitivo previo a grandes citas.

En cuanto al modelo de predicción, aunque la pérdida ListNet top-1 ha demostrado ser suficiente para alcanzar resultados competitivos, existen variantes más sofisticadas que podrían mejorar la calidad del ranking, especialmente al centrarse en los puestos que otorgan puntos UCI. En esta línea, la Tabla 10 resume cuatro alternativas ordenadas de menor a mayor complejidad: desde extender la propia ListNet a *top-k* hasta adoptar pérdidas basadas en optimizar directamente aproximaciones diferenciables de NDCG. Evaluar empíricamente estas opciones en próximas iteraciones permitiría cuantificar el compromiso entre rendimiento y eficiencia, y así refinar todavía con más exactitud la planificación de calendarios [27], [37], [38].

PÉRDIDA	COMENTARIO
ListNet *top-1* (actual)	*Baseline* rápido y estable.
ListNet *top-5*	Captura más del ranking, aún con coste bajo (5 × soft-max).
LambdaLoss, NDCG	Optimiza directamente un *upper-bound* de NDCG@K.
ApproxNDCG / SoftRank **[39]**	Suavizan NDCG con sigmoides, haciendo la computación algo más pesada.

Tabla 10. Potenciales funciones de pérdida.

6.4 Ética

A lo largo del desarrollo de este Trabajo de Fin de Grado se ha puesto especial atención en garantizar que tanto el uso de redes neuronales como de algoritmos evolutivos se lleve a cabo de manera ética y responsable. Una de las principales preocupaciones éticas abordadas es evitar el denominado sesgo de explotación, entendido como la tendencia de los algoritmos a favorecer excesivamente a ciclistas percibidos como más "rentables", asignándoles mayor número de días de competición. En consecuencia, se establece como esencial que la optimización algorítmica respete estrictamente los límites humanos relacionados con la fatiga acumulada, los periodos necesarios para una adecuada recuperación y, sobre todo, el bienestar físico y psicológico de los deportistas.

Asimismo, este trabajo defiende claramente que la inteligencia artificial no sustituye ni debe sustituir al cuerpo técnico, sino que se posiciona exclusivamente como una herramienta complementaria de asesoramiento. En este sentido, todas las soluciones generadas por el sistema son propuestas preliminares que deben ser validadas y aprobadas por profesionales cualificados, como el director deportivo o el personal médico del equipo. Aunque el aprendizaje automático es capaz de ofrecer alternativas eficientes para la planificación competitiva, la valoración definitiva siempre corresponde

al juicio humano, capaz de analizar factores cualitativos como posibles lesiones, el estado real de forma del corredor y las prioridades estratégicas a largo plazo del equipo.

El compromiso con una inteligencia artificial explicable e interpretable representa otro pilar esencial del enfoque ético adoptado en este trabajo. De acuerdo con autores como Barredo [40], es imprescindible que todos los usuarios puedan comprender de manera transparente cómo se generan las recomendaciones algorítmicas, qué variables son más influyentes y cuáles son las limitaciones inherentes al modelo utilizado. Esta transparencia es indispensable para que el personal de un equipo ciclista pueda realizar una evaluación crítica de las propuestas, identificar posibles sesgos o errores, y tomar decisiones informadas que integren la información algorítmica con su conocimiento profesional.

Por otro lado, se ha insistido especialmente en evitar cualquier tipo de discriminación algorítmica. El sistema desarrollado considera únicamente criterios deportivos objetivos, evitando cualquier posible sesgo hacia aspectos como la nacionalidad o características personales ajenas al desempeño deportivo directo.

No obstante, es crucial reconocer ciertas limitaciones inherentes al método propuesto. Las redes neuronales utilizadas están entrenadas exclusivamente con datos históricos, lo cual podría limitar su capacidad para anticipar mejoras repentinas de rendimiento. Por ejemplo, todos los ciclistas tienen saltos cualitativos durante su carrera deportiva, pero es prácticamente imposible saber cuándo se van a dar. Asimismo, los ciclistas jóvenes tienen diferentes procesos de adaptación al campo profesional, algunos rinden inmediatamente a un nivel excepcional, mientras que otros tienen un proceso de adaptación más paulatino.

Finalmente, cabe destacar que la implementación ética y responsable de estas tecnologías exige mantener un diálogo continuo entre desarrolladores, profesionales del deporte y deportistas. Esta colaboración multidisciplinar es clave para asegurar que la innovación tecnológica complemente el trabajo de los responsables deportivos y promueva el bienestar integral de todos los ciclistas involucrados.

A. Variables utilizadas

Lista completa de las variables utilizadas para entrenar los modelos:

Variable	Tipo	Descripción breve
rank	*target*	Posición final del corredor en la carrera (1 = ganador).
seconds_from_winner	*target*	Diferencia de tiempo respecto al vencedor, en segundos.
race_id	id	Identificador único de la carrera.
rider_id	id	Identificador único del corredor.
profilescore	continua	Índice global de dureza del recorrido.
vertical_meters	continua	Desnivel positivo total de la etapa (m).
race_ranking	continua	Como se compara la calidad de los inscritos respecto a otras carreras.
startlist_quality_score	continua	Calidad de los corredores inscritos.
total_distance_km	continua	Distancia total de la etapa (km).
min_elevation	continua	Altitud mínima del recorrido (m s.n.m.).
max_elevation	continua	Altitud máxima del recorrido (m s.n.m.).
mean_elevation	continua	Altitud media a lo largo de la etapa.
elevation_range	continua	Rango de altitudes (= máx – mín).
mean_latitude	continua	Latitud media del trazado.
mean_longitude	continua	Longitud media del trazado.
total_elev_gain	continua	Desnivel acumulado total (m).
longest_climb_km	continua	Longitud de la subida más larga (km).
dist_last_climb_to_finish	continua	Distancia meta-cima de la última ascensión (km).
total_km_climbing	continua	Kilómetros totales de ascenso dentro de la etapa.
total_climb_vertical_gain	continua	Desnivel total solo en secciones de subida (m).
stage_profile_score	continua	Puntuación compuesta del perfil completo.
final_25_score	continua	Dureza específica de los últimos 25 km.
total_climb_profile_score	continua	Puntuación agregada de todas las subidas.
total_elev_gain_last_1km	continua	Desnivel en el último km.
net_elevation_diff_last_1km	continua	Diferencia altitud inicio-meta en el último km.
sum_profile_scores_last_1km	continua	Suma de “profile scores” en el último km.
total_elev_gain_last_5km	continua	Desnivel acumulado en los últimos 5 km.
net_elevation_diff_last_5km	continua	Diferencia altitud en los últimos 5 km.
sum_profile_scores_last_5km	continua	Suma de “profile scores” en los últimos 5 km.
total_elev_gain_last_15km	continua	Desnivel acumulado en los últimos 15 km.
net_elevation_diff_last_15km	continua	Diferencia altitud en los últimos 15 km.
sum_profile_scores_last_15km	continua	Suma de “profile scores” en los últimos 15 km.
total_elev_gain_last_50km	continua	Desnivel acumulado en los últimos 50 km.
net_elevation_diff_last_50km	continua	Diferencia altitud en los últimos 50 km.
sum_profile_scores_last_50km	continua	Suma de “profile scores” en los últimos 50 km.
pct_distance_above_5pct	continua	% del recorrido con pendiente ≥ 5 %.
pct_distance_above_8pct	continua	% del recorrido con pendiente ≥ 8 %.
age	continua	Edad del corredor (años).
charac_i_plain	continua	Índice de rendimiento en llano
charac_i_mountain	continua	Índice de rendimiento en montaña.
charac_i_downhilling	continua	Habilidad bajando.
charac_i_cobble	continua	Capacidad en pavé.
charac_i_timetrial	continua	Aptitud en contrarreloj.
charac_i_prologue	continua	Rendimiento en prólogos.
charac_i_sprint	continua	Velocidad punta al esprint.
charac_i_acceleration	continua	Capacidad de aceleración brusca.
charac_i_endurance	continua	Resistencia aeróbica prolongada.
charac_i_resistance	continua	Tolerancia a la fatiga general.
charac_i_recuperation	continua	Recuperación entre etapas.
charac_i_hill	continua	Rendimiento en repechos cortos.
charac_i_baroudeur	continua	Tendencia a escapadas largas.
day_of_year	continua	Día del año (1–366) en que se disputa la carrera.
start_hour	continua	Hora local de salida (0–23).
last_3_avg_rank*	continua	Media del puesto en las 3 carreras previas.
days_since_last_race*	continua	Días transcurridos desde la última competición.
avg_rank_overall_previous*	continua	Media histórica de posiciones antes de la prueba actual.
last_5_avg_rank*	continua	Media de los últimos 5 resultados.
last_10_avg_rank*	continua	Media de los últimos 10 resultados.
last_3_rank_std*	continua	Desviación estándar de puestos en las 3 carreras previas.

*son eliminadas en las variantes “less features” de los modelos

Tabla 11. Lista y descripción de las variables utilizadas en modelado.

Referencias

[1] C. C. Ribeiro, «Sports scheduling: Problems and applications», *Int. Trans. Oper. Res.*, vol. 19, n.º 1-2, pp. 201-226, 2012, doi: 10.1111/j.1475-3995.2011.00819.x.

[2] G. Kendall, S. Knust, C. C. Ribeiro, y S. Urrutia, «Scheduling in sports: An annotated bibliography», *Comput. Oper. Res.*, vol. 37, n.º 1, pp. 1-19, ene. 2010, doi: 10.1016/j.cor.2009.05.013.

[3] D. Goossens y F. Spieksma, «Scheduling the Belgian Soccer League», *Interfaces*, vol. 39, n.º 2, pp. 109-118, 2009.

[4] F. Alarcón *et al.*, «Operations research transforms the scheduling of chilean soccer leagues and south American world cup qualifiers», *Interfaces*, 2017, doi: 10.1287/inte.2016.0861.

[5] K. Nurmi, D. Goossens, J. Kyng&aauml, y s, «Scheduling a triple round robin tournament with minitournaments for the Finnish national youth ice hockey league», *J. Oper. Res. Soc.*, vol. 65, n.º 11, pp. 1770-1779, 2014.

[6] G. Durán, M. Guajardo, y D. Sauré, «Scheduling the South American Qualifiers to the 2018 FIFA World Cup by integer programming», *Eur. J. Oper. Res.*, vol. 262, n.º 3, pp. 1109-1115, 2017.

[7] A. Dimitsas, C. Gogos, C. Valouxis, A. Tzallas, y P. Alefragis, «A pragmatic approach for solving the sports scheduling problem», en *Proc. 13th Int. Conf. Pract. Theory Autom. Timetabling, PATAT*, 2022, pp. 195-207. Accedido: 4 de junio de 2025. [En línea]. Disponible en: https://patatconference.org/patat2022/proceedings/PATAT_2022_paper_21.pdf

[8] «Hybridizing biogeography-based optimization and integer programming for solving the travelling tournament problem in sport scheduling | Request PDF», ResearchGate. Accedido: 5 de junio de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/390553092_Hybridizing_biogeography-based_optimization_and_integer_programming_for_solving_the_travelling_tournament_problem_in_sport_scheduling

[9] J. V. Imaicela Sarango y A. M. Ordóñez Mediavilla, «Desarrollo de una aplicación WEB que permita la gestión de reservaciones y generación automática de calendarios deportivos para la cancha sintética zona futbol.», 2012, Accedido: 5 de junio de 2025. [En línea]. Disponible en: http://dspace.unl.edu.ec/jspui/handle/123456789/14291

[10] O. Castrillán, «Ritmos cognitivos y algoritmos evolutivos en la programación de horarios universitarios», *Rev. Matemática Teoría Apl.*, vol. 22, n.º 1, pp. 135-152, jun. 2015.

[11] «Algoritmo genético», *Wikipedia, la enciclopedia libre*. 17 de marzo de 2025. Accedido: 5 de junio de 2025. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Algoritmo_gen%C3%A9tico&oldid=166174480

[12] G. Polyák y M. Poth, «Optimization of the Formula-1 Race Calendar Using Genetic Algorithm», en *2022 IEEE 20th Jubilee International Symposium on Intelligent Systems and Informatics (SISY)*, sep. 2022, pp. 000337-000342. doi: 10.1109/SISY56759.2022.10036315.

[13] M. Kyriakou, «Deep reinforcement learning for improving competitive cycling performance», *Expert Syst. Appl.*, oct. 2022, Accedido: 5 de junio de 2025. [En línea]. Disponible en: https://www.academia.edu/109727071/Deep_reinforcement_learning_for_improving_competitive_cycling_performance

[14] L. Kholkine, T. De Schepper, T. Verdonck, y S. Latré, «A Machine Learning Approach for Road Cycling Race Performance Prediction», en *Machine Learning and Data Mining for Sports Analytics*, U. Brefeld, J. Davis, J. Van Haaren, y A. Zimmermann, Eds., Cham: Springer International Publishing, 2020, pp. 103-112. doi: 10.1007/978-3-030-64912-8_9.

[15] M. Sagi, P. Saldanha, G. Shani, y R. Moskovitch, «Pro-cycling team cyclist assignment for an upcoming race», *PLOS ONE*, vol. 19, n.º 3, p. e0297270, mar. 2024, doi: 10.1371/journal.pone.0297270.

[16] «TrainingPeaks - Plan your training, track your workouts and measure your progress». Accedido: 12 de junio de 2025. [En línea]. Disponible en: https://app.trainingpeaks.com/

[17] «Strava | Aplicación de atletismo, ciclismo y senderismo: entrena, lleva un seguimiento y comparte», Strava. Accedido: 12 de junio de 2025. [En línea]. Disponible en: https://www.strava.com/

[18] L. Kholkine *et al.*, «A Learn-to-Rank Approach for Predicting Road Cycling Race Outcomes», *Front. Sports Act. Living*, vol. 3, oct. 2021, doi: 10.3389/fspor.2021.714107.

[19] «STGCN-LSTM for Olympic Medal Prediction: Dynamic Power Modeling and Causal Policy Optimization». Accedido: 5 de junio de 2025. [En línea]. Disponible en: https://arxiv.org/html/2501.17711v2

[20] N. Ahmad, «NTT Pro Cycling: Data-driven Innovation in Action», *NTT Tech. Rev.*, vol. 18, n.º 12, pp. 31-38, dic. 2020, doi: 10.53829/ntr202012fa4.
[21] D. Mlčoch y O. Hubáček, «Competing in daily fantasy sports using generative models», *Int. Trans. Oper. Res.*, vol. 31, n.º 3, pp. 1515-1532, 2024, doi: 10.1111/itor.13344.
[22] «Current weather and forecast - OpenWeatherMap». Accedido: 4 de junio de 2025. [En línea]. Disponible en: https://openweathermap.org/
[23] «PCS Profile score». Accedido: 4 de junio de 2025. [En línea]. Disponible en: https://www.procyclingstats.com/info/profile-score-explained
[24] *rapidfuzz/RapidFuzz*. (14 de mayo de 2025). Python. RapidFuzz. Accedido: 14 de mayo de 2025. [En línea]. Disponible en: https://github.com/rapidfuzz/RapidFuzz
[25] «Multilayer perceptron», *Wikipedia*. 12 de mayo de 2025. Accedido: 4 de junio de 2025. [En línea]. Disponible en: https://en.wikipedia.org/w/index.php?title=Multilayer_perceptron&oldid=1290081276
[26] A. Vaswani *et al.*, «Attention Is All You Need», 2 de agosto de 2023, *arXiv*: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.
[27] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, y H. Li, «Learning to rank: from pairwise approach to listwise approach», en *Proceedings of the 24th international conference on Machine learning*, Corvalis Oregon USA: ACM, jun. 2007, pp. 129-136. doi: 10.1145/1273496.1273513.
[28] L. Wu, C.-J. Hsieh, y J. Sharpnack, «SQL-Rank: A Listwise Approach to Collaborative Ranking», 6 de febrero de 2019, *arXiv*: arXiv:1803.00114. doi: 10.48550/arXiv.1803.00114.
[29] N. Kannen, Y. Ma, G. J. j. Van Den Burg, y J. B. Faddoul, «Efficient Pointwise-Pairwise Learning-to-Rank for News Recommendation», en *Findings of the Association for Computational Linguistics: EMNLP 2024*, Y. Al-Onaizan, M. Bansal, y Y.-N. Chen, Eds., Miami, Florida, USA: Association for Computational Linguistics, nov. 2024, pp. 12403-12418. doi: 10.18653/v1/2024.findings-emnlp.723.
[30] «Optuna - A hyperparameter optimization framework», Optuna. Accedido: 28 de mayo de 2025. [En línea]. Disponible en: https://optuna.org/
[31] A. Shahrour, «Optuna vs GridSearch», Medium. Accedido: 28 de mayo de 2025. [En línea]. Disponible en: https://medium.com/@abdalrahman_shahrour/optuna-vs-gridsearch-57227556c450
[32] K. Deb, A. Pratap, S. Agarwal, y T. Meyarivan, «A fast and elitist multiobjective genetic algorithm: NSGA-II», *IEEE Trans. Evol. Comput.*, vol. 6, n.º 2, pp. 182-197, abr. 2002, doi: 10.1109/4235.996017.
[33] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, y Q. Zhang, «Multiobjective evolutionary algorithms: A survey of the state of the art», *Swarm Evol. Comput.*, vol. 1, n.º 1, pp. 32-49, mar. 2011, doi: 10.1016/j.swevo.2011.03.001.
[34] «Eficiencia de Pareto», *Wikipedia, la enciclopedia libre*. 29 de noviembre de 2024. Accedido: 19 de mayo de 2025. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Eficiencia_de_Pareto&oldid=163841071
[35] «Feature Extraction and Personalized Sports Training for Athletes Using Variational Autoencoder (VAE)», en *ResearchGate*, feb. 2025. doi: 10.1145/3662739.3672309.
[36] D. P. Kingma y M. Welling, «Auto-Encoding Variational Bayes», 10 de diciembre de 2022, *arXiv*: arXiv:1312.6114. doi: 10.48550/arXiv.1312.6114.
[37] «Scale-Invariant Learning-to-Rank». Accedido: 12 de junio de 2025. [En línea]. Disponible en: https://arxiv.org/html/2410.01959v1
[38] L. Wang y T. Joachims, «User Fairness, Item Fairness, and Diversity for Rankings in Two-Sided Markets», en *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, jul. 2021, pp. 23-41. doi: 10.1145/3471158.3472260.
[39] M. Taylor, J. Guiver, S. Robertson, y T. Minka, «SoftRank: Optimising Non-Smooth Rank Metrics».
[40] A. Barredo Arrieta *et al.*, «Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI», *Inf. Fusion*, vol. 58, pp. 82-115, jun. 2020, doi: 10.1016/j.inffus.2019.12.012.